REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	10 Sep 97	

4. TITLE AND SUBTITLE
Automated Design and Optimization of Wire Antennas Using Genetic Algorithms

5. FUNDING NUMBERS

6. AUTHOR(S)
Derek S. Linden

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Massachusetts Institute of Technology

8. PERFORMING ORGANIZATION REPORT NUMBER
97-024D

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
THE DEPARTMENT OF THE AIR FORCE
AFIT/CIA
2950 P STREET
WPAFB OH 45433

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

14. SUBJECT TERMS	15. NUMBER OF PAGES	16. PRICE CODE
	143	

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

Standard Form 298 (Rev. 2-89) (EG)
Prescribed by ANSI Std. 239.18
Designed using Perform Pro, WHS/DIOR, Oct 94

Automated Design and Optimization of Wire Antennas Using Genetic Algorithms

by

Derek S. Linden

Submitted to the Department of Electrical Engineering and Computer Science on August 15, 1997 in partial fulfillment of the requirements for the Degree of Doctor of Philosophy in Electrical Engineering

ABSTRACT

A Genetic Algorithm (GA) has been used in conjunction with the Numerical Electromagnetics Code, Version 2 (NEC2) to create and optimize atypical wire antenna designs with impressive characteristics.

Antenna design parameters are encoded into an ordered series of numbers and/or symbols analogous to a biological chromosome. A cost function that quantifies how well a design meets the engineer's specifications is created. The GA uses these to generate and evaluate a population of designs. The most successful designs are then promoted and mixed through mating and mutation, while poor designs are removed. This process, difficult to trap in local minima, continues until convergence criteria are met, generally yielding excellent designs with no user intervention or initial guesses.

Three antennas have been optimized: a monopole loaded with a modified folded dipole, the Yagi antenna, and the crooked-wire genetic antenna.

Prior study of the loaded monopole had shown hemispherical coverage was possible. The GA found an asymmetric loaded monopole with an average variation in gain over the hemisphere of only 0.4dB, confirmed by measurement.

GA-optimized Yagi antennas surpassed the gain of conventional Yagis by about 1dB, improvement also confirmed by measurement. The GA designed a Yagi with a beamwidth of 50°-60°, sidelobes nearly 25dB down, and a 14% bandwidth—specifications difficult to achieve using conventional techniques.

The crooked-wire genetic antenna is several wires joined in series; locations and lengths are determined by the GA. Optimization for hemispherical coverage with right-hand circular polarization (RHCP) produced highly unusual shapes unrealizable using a conventional approach. RHCP hemispherical coverage was achieved with less than 4dB variation. Measurements verify the results.

GA performance was optimized and the effectiveness of real-numbered chromosomes was compared with the typical binary encoding for all antennas. Real-numbered chromosomes and a special mating process were found to give the same or higher quality designs and up to a 75% reduction in overall simulations.

This new process may revolutionize the design of wire antennas. It can be used to solve different problems of widely varying complexity, including those too difficult to attempt with traditional design methods.

Thesis Supervisor: Frederic R. Morgenthaler
Title: Professor of Electrical Engineering

Thesis Supervisor: Mark J. Jakiela
Title: Professor of Mechanical Engineering

Automated Design and Optimization of Wire Antennas Using Genetic Algorithms

by

Derek S. Linden

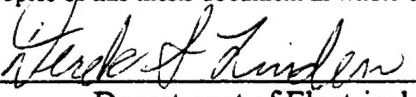
B.S. Applied Physics
United States Air Force Academy, 1991

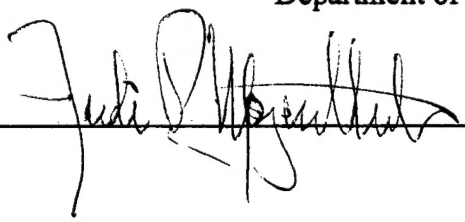
S.M. Electrical Engineering
Massachusetts Institute of Technology, 1993

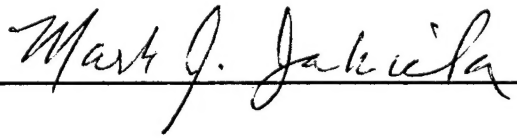
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

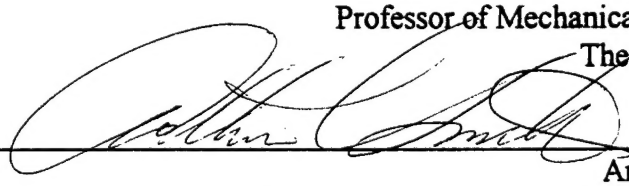
DOCTOR OF PHILOSOPHY IN ELECTRICAL ENGINEERING
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

SEPTEMBER 1997

Signature of Author: 
Department of Electrical Engineering and Computer Science
August 15, 1997

Certified by: 
Frederic R. Morgenthaler
Professor of Electrical Engineering
Thesis Supervisor

Certified by: 
Mark J. Jakiela
Professor of Mechanical Engineering
Thesis Supervisor

Accepted by: 
Arthur C. Smith
Chairman, Committee on Graduate Students
Department of Electrical Engineering and Computer Science

Automated Design and Optimization of Wire Antennas Using Genetic Algorithms

by

Derek S. Linden

Submitted to the Department of Electrical Engineering and Computer Science on August 15, 1997 in partial fulfillment of the requirements for the Degree of Doctor of Philosophy in Electrical Engineering

ABSTRACT

A Genetic Algorithm (GA) has been used in conjunction with the Numerical Electromagnetics Code, Version 2 (NEC2) to create and optimize atypical wire antenna designs with impressive characteristics.

Antenna design parameters are encoded into an ordered series of numbers and/or symbols analogous to a biological chromosome. A cost function that quantifies how well a design meets the engineer's specifications is created. The GA uses these to generate and evaluate a population of designs. The most successful designs are then promoted and mixed through mating and mutation, while poor designs are removed. This process, difficult to trap in local minima, continues until convergence criteria are met, generally yielding excellent designs with no user intervention or initial guesses.

Three antennas have been optimized: a monopole loaded with a modified folded dipole, the Yagi antenna, and the crooked-wire genetic antenna.

Prior study of the loaded monopole had shown hemispherical coverage was possible. The GA found an asymmetric loaded monopole with an average variation in gain over the hemisphere of only 0.4dB, confirmed by measurement.

GA-optimized Yagi antennas surpassed the gain of conventional Yagis by about 1dB, improvement also confirmed by measurement. The GA designed a Yagi with a beamwidth of 50°-60°, sidelobes nearly 25dB down, and a 14% bandwidth—specifications difficult to achieve using conventional techniques.

The crooked-wire genetic antenna is several wires joined in series; locations and lengths are determined by the GA. Optimization for hemispherical coverage with right-hand circular polarization (RHCP) produced highly unusual shapes unrealizable using a conventional approach. RHCP hemispherical coverage was achieved with less than 4dB variation. Measurements verify the results.

GA performance was optimized and the effectiveness of real-numbered chromosomes was compared with the typical binary encoding for all antennas. Real-numbered chromosomes and a special mating process were found to give the same or higher quality designs and up to a 75% reduction in overall simulations.

This new process may revolutionize the design of wire antennas. It can be used to solve different problems of widely varying complexity, including those too difficult to attempt with traditional design methods.

Thesis Supervisor: Frederic R. Morgenthaler
Title: Professor of Electrical Engineering

Thesis Supervisor: Mark J. Jakiela
Title: Professor of Mechanical Engineering

Table of Contents

List of Figures

List of Tables

Acknowledgments

The funding for my education at MIT was sponsored by a fellowship from the Fannie and John Hertz Foundation. I greatly appreciate the support they provided, without which this work would not have been possible. I thank them for their long-term commitment, as they provided tuition and book allowances for five full years!

I would like to thank the management at Rome Laboratory who supported me in my research, particularly Dr. Steve Mittleman, who allowed me to research this topic and pursue it at MIT while I was in his branch.

I gratefully acknowledge the support of Col Head and Col Enger of the Physics Department at the U.S. Air Force Academy, who sponsored my last year at MIT, allowing me to have full-time student status that enabled me to finish my degree. Thanks to the excellent staff at AFIT, especially Maj Conejo and Maj Tolle, who managed my program.

Thanks to Lt Col Randy Haupt, who introduced me to Genetic Algorithms and their application to electromagnetics. I very much appreciate his sparking my interest in this fascinating field of research and providing the tutorial which helped me get my first GA program on-line.

I am particularly grateful to the members of my thesis committee, Prof. Frederic Morgenthaler, Prof. Mark Jakiela, Dr. Ed Altshuler, Prof. David Staelin, and Dr. Robert Shin. Thanks to each one for consenting to be involved and especially for putting in the time and effort to attend meetings, read and critique the draft of this thesis, and have discussions with me about the work. Thanks to them for working so well together, even though their backgrounds and fields of interest differed greatly.

Thanks to Prof. Morgenthaler and Prof. Mark Jakiela for consenting to be co-supervisors. Prof. Jakiela provided excellent guidance regarding GAs, and I appreciate his involvement even though as a mechanical engineer antennas were not familiar territory. I especially appreciate his continued support after he moved to Washington University in St. Louis well before the thesis was finished. Thanks also to Prof. Morgenthaler for his continued support and involvement after his retirement last year.

I would especially like to thank Dr. Ed Altshuler, who introduced me to wire antenna design and NEC2, and who has been my co-author on every paper but one regarding this work. He has been a constant source of help and guidance, and his labor has allowed this research to proceed much further and gain much more acceptance in the antenna community than I would ever have been able to accomplish alone. He has been a superb colleague to work with, and I appreciate the symbiotic relationship we have shared: he providing vast experience and knowledge of antennas, while I provided the GA expertise. He has done much analyzing, fabricating and measuring of the antennas I produced, and he has had excellent ideas on how to pursue better designs. I appreciate the time we have worked together, and I acknowledge the great contribution he has made to the research contained in this thesis.

Thanks to my Dad & Mom, who provided encouragement when the path seemed ominous, who helped me through the rough spots, and kept my perspective within the bounds of reason.

Heartfelt thanks also to my wonderful wife, Rebecca, who encouraged me, listened to me pontificate about the vagaries of the GA and antennas, helped me sort through my ideas, made excellent suggestions, and even helped me create illustrations and edit this thesis.

But most of all, thanks to God, who not only loaned me the talent and provided the support and circumstances which made it possible to finish this thesis, but also created the amazing and complex system of genetics at the Creation. At the root, it is He who is responsible for all the foregoing acknowledgments. Thanks to Him for creating such a fascinating, complex world in which to do research. And, especially, thanks for His Son, Jesus, though whose death and resurrection He has provided for me an eternity spent with Him, a future infinitely better than I deserve.

Chapter 1: Introduction

1.1 Current methods of antenna design

In 1887, Hertz demonstrated the existence of electromagnetic waves experimentally and confirmed Maxwell's predictions. Hertz used copper wires end-loaded with large spheres which were excited by a spark discharge, which launched an electromagnetic wave. In 1898, Braun designed a circuit consisting of a spark gap in its primary circuit which was inductively coupled to a linear antenna in the secondary circuit, creating the first wire antenna.

Since wire antennas first appeared, a variety of useful configurations have emerged, like the dipole and its counter-part monopole over a ground screen, the rhombic, Beverage, Yagi, log periodic, loop, helix, and spiral antenna. Though there are many different designs, they have all been designed and optimized using a similar approach.

This approach used by antenna engineers generally limits antennas to relatively simple structures. If the engineer wishes to create a new antenna, either to find a better solution to a particular problem or simply for the sake of creating a new antenna, he or she will often use an elegant or intuitive geometrical arrangement that exploits some electromagnetic property. There are two major ways to arrive at this new design: start from basic theoretical equations that lead from the desired qualities to the form of the antenna that produces those qualities, or take existing designs and combine parts of them into a single design. However, if one takes the mathematical approach, one is limited to simple structures because electromagnetic analytical expressions quickly become very involved for even moderately complicated structures. If one combines existing designs or takes a more intuitive approach, one is left without much guidance in the proper optimization of the design. In addition, it is difficult to know if such a design is going to work well when optimized.

In most cases, an engineer will not wish to create a new design, but rather optimize an existing one for a particular situation. In this case, an engineer finds an existing configuration that may have the desired electromagnetic characteristics once optimized. He or she works with appropriate equations to determine initial guesses at its proper dimensions and parameters, and uses an electromagnetic simulator or an analytical expression to predict its performance. If the performance is not acceptable, the antenna is redesigned, using guides such as intuition, experience, approximate equations, or empirical studies to determine which parameters to change to improve performance. While this design technique has produced many different antennas that perform adequately, it is unlikely to produce truly optimum results if there are more than a couple unknowns. It also requires a familiarity with many different designs, and enough experience so an acceptable solution can be reached in a reasonable amount of time.

When the high speed digital computer became available in recent years, it became possible to analyze more complex wire structures in shorter time spans. While the simulator is faster than using pen and paper or a calculator, there are too many variables in even simple designs for a person to keep track of effectively.

The speed of the simulator and the relative ease of modifying and resimulating a design leads to an interesting problem. Since one need not understand all the underlying physical equations of the structure any longer, and can simply tinker with the design on the computer, "folklore" can result: certain factors are seen to affect certain antenna characteristics in certain ways, yet they may or may not. This problem is not unique to computer simulators, though—folklore can develop in actual electromagnetic testing situations as well. For some of the more complicated aspects of electromagnetics, even years of experience may not result in useful intuition regarding them.

The art of antenna design, while being a fairly slow process, is also mature—people have been designing antennas since the turn of the century, and there are hundreds available. Engineers have been tweaking these designs for some time, occasionally finding new ones, and using the current body of knowledge as a basis for further work. This limited search method—which can be likened to using a few thousand well-tuned neural nets and expert systems in search of better antennas after many years of prior searching—provides relatively slow progress.

There are a few antenna optimization programs that are appearing on the market. In most, if not all, cases, the general shape of the wire antenna is still predetermined and the individual wires and parameters that constitute that particular

design are optimized. However, these programs are limited in effectiveness because they use gradient-based optimization approaches. These types of approaches require a well-behaved solution space and/or a good initial guess to arrive at good optimized designs, because they are sensitive to local minima and are trapped very easily. This means the engineer must still use intuition and initial guesses to optimize an antenna.

In addition, the standard design cycle limits the types of designs that are tried to those that have an intuitive or mathematical logic about them. Symmetry is often present, and structures are kept relatively simple to allow for easier understanding, analysis and modification. Nearly all of the designs produced by engineers have the characteristic of "making sense" when one looks at them or their equivalent circuit schematics. Most all of them look like they should work. However, non-intuitive configurations can sometimes work as well as or better than intuitive ones. As an example, note that people use bent coat hangers, aluminum foil, and antennas in strange-looking configurations to get good radio or TV reception, unhindered by knowledge of theory and guided only by monitoring reception quality—a very simple form of random-walk optimization. It is therefore of interest to find a way to search for such counter-intuitive solutions using a well-validated, standardized optimization methodology. One technique in particular, the Genetic Algorithm (GA), is sufficiently powerful to search this kind of counter-intuitive solution space, as will be shown in this thesis.

1.2 Motivation for automated design procedures

Government and commercial systems employ thousands of different types of wire antennas for communications and remote control, radar, and surveillance. All of these systems employ antennas that are limited by the above method of design optimization, but even so most of these systems perform adequately with these antennas. However, as communications and electronic sensors rise in importance in an increasingly technological and interconnected world, there will grow an expanding need for high-performance, customized antennas that are optimized for performing specialized functions. However, as mentioned above, the current method of antenna design is experience-intensive, and there is no very fast or automated way to design the special antennas they need. There is a growing demand, therefore, to find a faster, less labor-intensive way to design antennas.

For instance, indoor communications have grown in importance through the use of radio and cordless phones. Though most hand-held transceivers currently use linearly polarized antennas, it would be advantageous for an antenna to have dual polarization so that incoming signals that have become depolarized due to multiple reflections from walls and fixtures can still be detected. Experiments with monopole antennas loaded with modified folded dipoles [1,2] or loops [3] have shown that it is possible to achieve near-hemispherical coverage with these configurations. The monopole radiates a vertically polarized wave that provides coverage at the lower elevation angles while the folded elements or loop radiate a horizontally polarized wave that provide coverage at the higher elevation angles. Thus, the antenna provides dual polarization coverage at medium elevations, though it is still linearly polarized at high and low elevations. However, this antenna with six unknowns needs to be optimized before it will serve this purpose effectively.

As another example, earth-to-satellite communication and navigation systems are becoming more and more commonly used and low-cost. These systems will soon require ground-based antennas that are circularly polarized and have near-hemispherical coverage. Circular polarization is necessary for systems operating at frequencies below 3 GHz since the Faraday rotation produced by the ionosphere can cause a linearly polarized wave to rotate out of alignment with the receiving antenna. In a worst case scenario, the incoming wave becomes cross polarized so that no signal is received. A circularly polarized signal eliminates this problem. Near hemispherical coverage is also desirable since the earth-based antenna is often required to receive a signal from a satellite anywhere above the horizon, except at low elevation angles. (At low angles, signals have multipath components that can disrupt system performance.) Currently, helical or patch antennas are used for this application, but these antennas are generally narrow-band and require a phasing network, which increases their complexity and cost. In addition, these antennas are usually somewhat directional, meaning they need to be pointed more precisely toward the satellite to be used effectively—an inconvenience for mobile systems.

In order to study primeval hydrogen at the edge of the known universe using the Arecibo radio telescope, there is a need for a special feed antenna, with a 60° beamwidth, 14% bandwidth, linear polarization, and -25dB sidelobe level.

Helix antennas have been used, but with unsatisfactory results, and there does not appear to be any conventional antenna that meets their requirements.

Each of the above problems will be explored and solved in this thesis. However, there are many more of these antenna problems waiting to be solved. The tracking of hospital patients, biomedical research, applications requiring ultra-broadband antennas, remote sensing, electronic warfare applications, and many others, are all demanding antennas that meet their needs. Meeting them rapidly and effectively will require a new approach to antenna design, because the current methods are too limited to keep pace with the rising demand. What is needed is an automated design system that can search areas of antenna design heretofore unsearchable, and solve antenna design problems unhindered by the limits of human intuition and experience. This need is the motivation behind this thesis.

1.3 Previous work

Both a literature search and an excellent review article of GAs as applied to electromagnetics [4], with papers listed up to February of 1997, have shown that there is relatively little previous work on automated antenna design. Out of roughly 70 papers on the optimization of electromagnetic designs since 1990, only 29 involve antennas. Of the 29, 18 involve antenna arrays.

Haupt [5] describes how to use GAs to design thinned antenna arrays. An antenna array is a pattern of small, simple antenna elements that are operated with various phase differences between them to allow electronic steering of the radiation pattern. Most such antennas have regularly spaced antenna elements, however, it is often desired to remove some elements to achieve even better radiation patterns and allow for a cheaper antenna array (fewer elements means lower cost). However, it is difficult, without an exhaustive search, for a person to know which elements to remove. A technique like a GA is ideal for such a search. Haupt uses a specific type of antenna--a planar, regular geometric array—where elements are kept or removed to determine the characteristics of the antenna. He has also done similar work in backscattering from a grid and various absorption problems [6]. The designs are well-defined (strips of wire or absorbent material that are either present or absent) and specific to one application. The problem of designing robust antenna-array processors (not an antenna itself, but what controls the elements in an array) using quadratic programming techniques is described in [7]. There is automated design here, but only for this specific problem of processor design. The other array papers follow similar paths with GA optimization of array thinning, amplitude or phase tapering, specified nulls and designing arrays with specific radiation patterns. Thus, the majority of the antenna work available in the literature is focused on arrays of antennas that are easily "plugged in" to a GA.

The remaining 10 papers that involve antennas focus on single-element designs. 5 of them involve the design of ultra-broadband antennas using GAs [8-12]. A number of RLC loads are placed along monopoles or two-element arrays of monopoles, with the RLC values and the load locations to be determined by the GA. The results of such GA optimization are very good in these papers.

References [13] and [14] regard using optimization techniques and special simulation techniques for the design of reflector antennas, which are widely used and not necessarily easy to design. They have focused on general design rather than specific situations, still allowing the designer to determine and place the parts of the antenna.

The remaining three papers have resulted from the research for this thesis [15 - 17].

Regarding other optimization techniques applied to wire antennas, Landstorfer and Sacher [31] optimized dipoles and monopoles for effectiveness as stand-alone antennas and as elements in Yagi antennas using gradient and simplex searches. Their curvilinear antennas, though not quite as radical as those shown in Chapter 6, are very unusual and effective. However, since they were working prior to 1985, the complexity and application of their designs was limited by the computer power available at the time. Therefore, their optimization of monopole (and equivalent symmetric dipole) elements focused strictly on maximizing effective height (and hence directivity), given a wire length. Their Yagi antennas with curvilinear elements, optimized for directivity and/or sidelobe attenuation, were based on the optimization of one element for a particular purpose, the rest of the antenna being determined by rules based on the shape of that element. The single-wire antennas and Yagi elements were constrained to be two-dimensional. The variables were angles of bends at points equispaced along the wire. In addition, the initial guess was simple since the starting antenna was a straight-wire monopole. In each of their optimizations, the authors were

forced to simplify their approaches using electromagnetic theory and minimize the variables involved to reduce their search spaces to a manageable size.

1.4 The Purpose of the research

The reason for this research to be done is to explore an area not much considered since [31]: automating the optimization of antenna shapes and configurations themselves. For reasons listed in Chapter 2, the research has been limited to wire antennas. This research not only looks at more conventional design problems, but expands the horizon of antenna design to very loosely constrained search spaces that have no set theory of operation before the optimization takes place. As will be shown, the GA is sufficiently powerful to search much more complicated and difficult spaces than those of [31].

The purpose of the thesis itself is to inform the reader of as much of what I have learned over the course of this research as possible, especially regarding how to apply GAs to the complicated, time-consuming problems that constitute antenna design. It is intended to be practical in nature, and hopefully will impart much of the experience I have gained to those who will further this work. If, after reading this thesis, the reader can begin his or her own work where I left off, I will have succeeded.

Chapter 2: Basic Antenna Concepts

As the reader may not have a background in electromagnetics, this chapter describes some of the concepts and terms that will be used in the analysis of antenna designs and their characteristics. Some of the key qualities of antennas, especially wire antennas, will be described in practical, application-oriented terms.

An antenna is a device which provides a means for radiating or receiving radio waves. It is a transducer, transforming a signal from a transmission line, e.g., a waveguide, a coaxial cable or even a pair of wires, into an electromagnetic wave propagating (i.e., traveling) in free space with the desired properties [33]. (It should be noted that, though only a vacuum is truly "free space," normal air is very close to free space in its electromagnetic properties, with the notable exception of the ionosphere.) Antennas also perform the reverse function: transforming an electromagnetic wave in free space into a signal on a transmission line.

This transformation is necessary, because electronic signal generators, receivers and processors almost never use signals internally that are ready for free-space propagation. Electromagnetic signals inside these devices are constrained and directed using transmission lines that have significantly different impedance properties from free space. As will be discussed in Section 2.2.2, this mismatch at the junction between the transmission line and free space means that simply opening these transmission lines to free space would result in a very small propagating wave.

Not only does an antenna provide the proper impedance transformation for power transfer, but it also provides control over the directional properties of the propagating wave, allowing signal power to be aimed in the desired directions and with the desired polarization. Both of these qualities are important for ensuring a transmitted signal is able to reach its destination with sufficient power and be received with maximum efficiency by the intended receiver or receivers.

This chapter will discuss the wire antenna class, electromagnetic wave basics, radiation patterns, antenna impedance, frequency dependence, antenna measurements, and will introduce the Numerical Electromagnetics Code, Version 2 (NEC2) [18].

2.1 Antenna classes

There are many kinds of antenna classes, such as reflector antennas (e.g., dish antennas), phased array antennas consisting of regularly spaced multiple elements, wire antennas, horn antennas, and microstrip and patch antennas which are printed on metal-clad dielectrics (e.g., circuit boards). Each of these classes use different structures and exploit different properties of electromagnetic waves.

This thesis is limited to the class of wire antennas. There are a few reasons for choosing wire antennas over other classes: first, the simulator of choice, the Numerical Electromagnetics Code, Version 2 (NEC2), as discussed at the end of this chapter, is a fast, accurate, widely-used, public-domain wire antenna simulator. Second, wire antennas are easy and inexpensive to build and test, and thus there is a cost-effective way to validate the results of an optimization. Third, wire antennas are a particularly flexible class—they can be used to solve many different types of design problems: high or low gain, broad or narrow band. Other classes, like horns and reflectors, are much more restricted in the types of problems which they can solve.

A wire is defined as a conductor (e.g., a piece of metal) whose length is much greater than its diameter. Thus, an antenna is a wire antenna if it is composed primarily of one or more wires. The wires do not have to be connected. Unconnected elements are called parasitics, and act as passive antennas that receive radiation from the surrounding connected elements and re-radiate them in different directions. Such parasitics can produce excellent results, as the Yagi antenna to be covered in Chapter 5 will show.

A wire antenna can include objects that are not wires as well. Resistive and reactive (inductive and/or capacitive) circuits can be added to the structure, which cause changes in the operation of the antenna, and can change its

impedance, make it more or less sensitive to frequency, etc. The use of such loads was not explored in this thesis, however.

Another non-wire object used in wire antennas is a ground plane—at its simplest a large, flat piece of metal underneath the antenna. It is often used in conjunction with a wire antenna, and acts as an electromagnetic "mirror" for the antenna, changing the antenna impedance and directional properties. A ground plane can decrease an antenna's required height, simplify its the construction, and help direct signal power. In addition, the roof of a car or fuselage of an airplane is a type of ground plane, and antennas that will be affixed to such places need to be designed for use with one. Single-wire antennas fed at their base, like a monopole (e.g., a car antenna), generally use a ground plane, since transmission lines that feed wire antennas have at least two conductors, one of which connects to the antenna while the other connects to the ground plane. Otherwise, the connection would only include one wire of the transmission line, a situation which generally hampers performance. As will be seen, the loaded monopole and crooked wire antennas use ground planes, and the Yagi antenna can use one, though it is not common for this type of antenna.

Wire antennas can be excited, i.e., fed the desired signal, in many different ways. As mentioned above, a single-wire antenna over a ground plane can be fed most simply at its base, for example, with the center of a coaxial line connected to the monopole, and the outer coaxial shielding connected to the ground plane. It is also common and relatively easy to feed a wire antenna using a balanced, two-wire line, like the two-wire cable that runs between a television set and a pair of "rabbit ears." In this thesis, all the antennas are fed using one of these methods. Though there are more complex ways to excite antennas with multiple sources with differences in amplitude and phase between them, doing so adds significantly to the cost and complexity of the antenna. Therefore, it was chosen to simplify the feed structure and concentrate on the antenna design beyond the feed point to improve performance.

The next section discusses electromagnetic waves and general antenna properties common to all antennas. Because the research was limited to wire antennas, however, when specific examples or instances arise they will be discussed in the context of wire antennas.

2.2 Electromagnetic waves and antennas

Maxwell's equations govern the electromagnetic waves generated and received by all antennas, wire or not. These equations are listed below.

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t}$$

$$\nabla \cdot \mathbf{D} = \rho$$

$$\nabla \cdot \mathbf{B} = 0$$

Because there are only four of them, with only two or three terms each, Maxwell's equations appear deceptively simple. Even problems with only a small number of elements can be very difficult to solve analytically, and simple-looking antennas can exhibit very complicated behavior.

Though they are all governed by these four equations, different types of antennas transform a signal into propagating waves in unique ways, and each has distinct characteristics. Some are very directional, sending and receiving a signal primarily in one direction, while others send and receive in nearly all directions at once. The impedances at the antenna feed points and the polarization of the waves they transmit can be very different. Some only operate over a small band of frequencies, while others can operate over a wide frequency range. Each of these qualities: radiation pattern, polarization, impedance, and frequency dependence will now be discussed in turn.

2.2.1 Radiation pattern and Gain

Before discussing radiation patterns, it should be noted that whether an antenna is used to transmit or receive, all its characteristics are the same. This is the property of reciprocity. If an antenna transmits, it has the same directivity, polarization, and impedance as when receiving a signal. Thus, though one may simulate an antenna's performance while transmitting, one can measure it while it is receiving and the performance will be the same.[1] Though the discussions below will focus on the radiation patterns of antennas while transmitting or receiving only, keep in mind the antenna characteristics for the other action are the same.

In three-dimensional space, there are two angles required to specify a direction: the elevation θ and the azimuth ϕ. Figure 2.1 shows these angles on a set of axes.

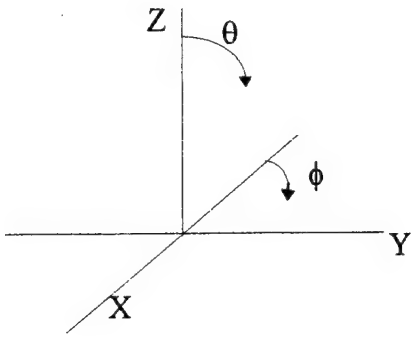


Figure 2.1. θ and ϕ on a 3-D axis system. Arrows begin where NEC2 defines 0 degrees for θ and ϕ.

An important characteristic of antennas is their directional nature. There is no electromagnetic antenna that is a completely isotropic (i.e., omnidirectional) radiator, though some can approach this limit [34]. Gain is a quantity used to relate an antenna's directional properties to this theoretical isotropic radiator.

In the case of a lossless antenna, gain is the ratio of power density being transmitted by an antenna in a particular direction to the average power density being transmitted in all directions. The average power density of any lossless radiator is $P_o/4\pi r^2$, where P_o is the total power input into the antenna, and r is the distance between the point of observation and the location of the radiator. This is so because if one encircles the radiator with a sphere of radius r, the total amount of power that can radiate through that sphere is P_o. Since the surface area of the sphere is $4\pi r^2$, the average power is given by $P_o/4\pi r^2$. So:

$$\text{Gain} = S_r(\theta,\phi)/ (P_o/4\pi r^2),$$

where $S_r(\theta,\phi)$ is the radiated power density as a function of angle, averaged over time and in the direction of r [20].

The average power density $P_o/4\pi r^2$ is exactly the amount that an isotropic radiator would send in all directions, so gain can be thought of as a comparison between an antenna and an isotropic radiator. An isotropically radiating antenna, then, would have a gain of 1 in all directions [20].

When loss is introduced into an antenna the gain decreases, because P_o is not the power radiated, but the power input into the antenna. Resistive loss turns some of that power into heat, reducing the radiated power and decreasing the radiated power density. However, the isotropic radiator's power density remains $P_o/4\pi r^2$. It is still of interest to compare the radiated power density to an isotropic radiator with the same radiated power. Hence, directivity is a quantity that is often used, and it is:

[1] For this property to hold, however, the materials in the antenna must themselves be reciprocal. This is almost always the case as the metals and dielectrics typically used in antennas are reciprocal. However, some nonreciprocal materials like magnetized ferrites can be used to make specialized antennas that have transmitting and receiving properties that differ. These antennas are uncommon and were not considered in this research.

$D(\theta,\phi) = S_r(\theta,\phi)/ (P_{radiated}/4\pi r^2)$.

It is related to gain by:

$D(\theta,\phi) = G(\theta,\phi) / (P_{radiated}/P_o)$.

However, in the lossless case, gain and directivity are the same [20].

Gain versus angle can be placed in a chart called a radiation pattern. An example pattern is shown below.

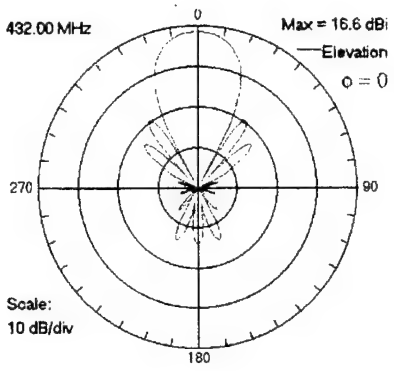


Figure 2.2. Example radiation pattern.

The gain in antenna patterns are usually in units known as decibels (dB), which relates to a ratio of power or power densities by the following expression: $dB = 10\log_{10}(P_1/P_2)$. Decibels are used in many different applications, like to quantify the power of a signal in a transmission line, or the strength of acoustic waves, as well as gain. In these applications, P_2 is usually a standard amount, like a milliwatt in the case of the transmission line signal. To indicate the value of P_2 in a particular case, an extra letter is added to dB, for instance dBm to indicate that P_2 is 1 mW. In the case of gain, P_2 is the power density of the isotropic radiator. The abbreviation dBi thus refers to gain compared with an isotropic radiator. However, the "i" is sometimes left off, and is understood from context.

Using decibels smoothes the variations in the pattern, and the logarithmic scale provides more relevant insight to antenna performance than a graph of linear power variation. A loss of 3dB reduces the power by half, while a loss of 20dB means a 100-fold loss in power.

Note that the figure above shows a very directional pattern. One small range of angles shows the greatest amount of gain. This lobe on the pattern is called the main lobe. This lobe generally has the highest gain in the pattern, and the maximum gain of this lobe and its width are often factors to be optimized. In a more uniform radiation pattern, there is usually only one lobe, but in a directive pattern as above where the main beam is desired to cover only a small portion of the pattern, there can exist other lobes. These other lobes are called sidelobes, and usually the designer seeks to minimize them. At the angles between the lobes are nulls, or directions where the antenna does not transmit any power.

Beamwidth is a term used to describe the angular span of the main lobe. It is most often measured from the maximum part of the lobe to points on either side that are 3dB lower in gain, i.e., where the signal has lost half its power. It should be noted that signals are often still receivable with even less gain, depending on the situation. Thus, the half power beamwidth may not be as relevant as the term seems to imply, as a signal variation of even 4dB can be very usable and operate as if the pattern were essentially flat for some engineering purposes.

However, a dip of more than 5-10dB is usually considered a serious decrease in power. An amplifier can only handle a certain amount of dynamic range before it can no longer amplify the weak signals enough and still remain linear when a strong signal comes through, and areas where signal cannot be received will start to develop in the pattern if

the dips are too large. In some cases this is desired, as when an antenna is to have only a narrow field of view, as in the case above. In other cases, as when a broad, even beam is required, this variation can be a problem. Such a beam is shown below.

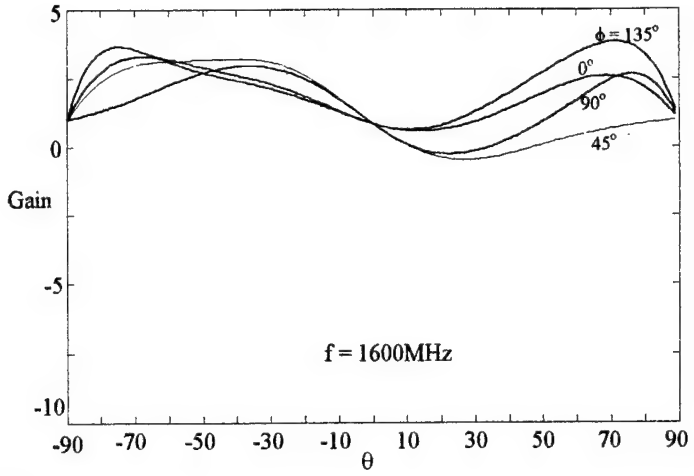


Figure 2.3. Another example radiation pattern.

The pattern above is rather uniform, in contrast to the first example. Notice that it is a rectangular plot instead of a polar plot like the first example pattern. The rectangular plot emphasizes the size of the peaks and valleys, which would be difficult to see on a polar plot, and is more revealing for less-directive antennas. For directive antennas, the polar plot gives one a better feel for the behavior of the antenna. The X axis is elevation, and curves are shown for four azimuthal angles. These types of curves are often called "cuts," as they are analogous to "cutting" a hemispherical pattern along a given azimuth angle and looking at the gain variation along the edge of the cut.

Before closing the discussion on radiation patterns and gain, it should be noted that, for many applications, the antenna engineer may wish to make an antenna small yet very directive. However, there is an upper limit on how directive a real antenna can be compared to its size—a limit that may not be evident in simulation.

A radiation pattern that is to be uniform, or nearly so, requires only a small antenna. But if an antenna is desired to be very directive, that is, have over a few dB of gain, it will need to be several wavelengths in size. However, it is possible for a simulation of a small antenna to show high gain, and if it does, that antenna is probably in a mode known as supergain.

In supergain, large electric and/or magnetic fields that do not generate propagating waves but store large amounts of energy are created near the antenna, and if the wires are simulated as perfect conductors, these fields exist without causing energy loss. Such fields allow the antenna to have a high gain, even though it occupies a small volume. Unfortunately, real materials do not conduct perfectly, and, beyond a certain point, these high fields are not possible due to resistive loss.

A theoretical limit for supergain is described in [19] as follows: if the antenna's largest dimension is denoted by 2a, then the gain must be less than $4a / \lambda$. If the gain is higher, the antenna is in supergain.

It should be noted that it is possible for real antennas to have a modest amount of supergain. For instance, arrays of resonant elements like a Yagi antenna can be in supergain, and still realizable. The 5.16λ (λ being the standard symbol for wavelength) Yagi antenna that was built and tested in Chapter 5 shows 17dB of gain, but according to the expression above, its dimensions limit it to a maximum gain of 10.37, or 10.16dB. Thus, it is clearly in supergain. However, beyond a certain amount of supergain, even these structures cannot achieve the high fields required. Thus, when optimizing an antenna for maximum gain, the possibility that a promising antenna is in supergain should be considered by the designer before fabrication and testing.

2.2.2 Impedance

In addition to the radiation pattern, the impedance of an antenna is a very important design consideration. In order for power to flow most effectively, with the smallest amount of reflection, the impedances of two structures that are connected to one another should be identical. If they are not, it will be more difficult for power to flow from one of the objects to the other.

Take for instance the interface between water and air—two materials with different electromagnetic impedances. Though they are not conductors, electromagnetic waves (like visible light) travel through them, and they each have a different impedance. Regardless of how clean and clear the water is, a reflection can be seen in it when one looks at it from air. Similarly, when one is underwater, one will see a reflection at the interface of water and air.

Metal electromagnetic components get their characteristic impedance from their configuration and from any dielectrics used in their construction. They can have both real impedance (resistance) and imaginary impedance (reactance). This impedance is most important at the points where connections are made to other devices, like the wire that feeds the signal to the antenna, and thus simulators like NEC2 give the impedance of the antenna at such points.

When two electromagnetic devices with different impedances are connected, reflections occur at the interface, and a standing wave will result. A standing wave is a combination of the original wave and the reflected wave, each of which is moving along the transmission line in opposite directions. The interference of these two waves affects the wave envelope. The envelope is the line that traces out the maximum voltages that appear at any time at each point along the line. Thus, the envelope is position dependent only. If there is no mismatch, and therefore no reflected wave, this envelope is a flat line. As the signal wave travels along the transmission line, each point will at some time contain a wave peak, and thus all points on the line will have a maximum voltage equal to the signal wave's maximum voltage. The presence of a reflected wave causes the envelope to develop peaks and valleys, as some points on the line will always be affected by constructive interference and others by destructive interference, augmenting and decreasing the envelope at each point [20].

Voltage Standing Wave Ratio, or VSWR, is a way to quantify this reflected-wave interference, and thus the amount of mismatch at the junction. VSWR is the ratio between the highest voltage and the lowest voltage in the signal envelope along a transmission line [20]. If the VSWR is high, there is a great deal of interference and a significant mismatch. If it is low, the interference is low and the match is good. A VSWR of 3.0 or less is considered adequate for many low-power applications, while a VSWR less than 1.5 or 2.0 is desired if power considerations are important. A VSWR of 1.0 is a perfect match, as the envelope is a flat line, implying the maximum and minimum voltages are equal. Of course, VSWR can never be less than 1. VSWR is easy to measure, and as it is a common parameter specified by antenna designers, it is often an important quantity to optimize.

As described above, the mismatch, and thus VSWR, are dependent on the impedance of the transmission line feeding the antenna as well as the antenna itself. One must know the impedance of both parts of the junction, then, in order to calculate VSWR. The most common transmission line impedance is 50 ohms—the impedance of standard coaxial cables—though other impedances can be used, and often are. For example, an impedance of 300 ohms is used for two-wire television antenna cables.

Besides making sure that power is transmitted efficiently, there is another related reason to be concerned with VSWR. If there is a driving circuit, say an amplifier, that will have only a fraction of its power actually used in the transmission of radio waves, the rest will be carried by the reflected wave back into the system, to eventually be absorbed by its resistive components, producing heat. This can damage electronic components. To prevent this damage, some high powered microwave amplifiers, like traveling-wave tubes, will not transmit if a high VSWR is detected.

2.2.3 Polarization

Electromagnetic waves are composed of two components: an E-field (electric field) component and an H-field (magnetic field) component that exists at right-angles to the E-field following the right-hand rule ($\mathbf{E} \times \mathbf{H} = \mathbf{k}$—a vector in the direction of propagation), and whose magnitude is proportional to the magnitude of the E field. Thus,

the wave is asymmetric, and has a definite orientation. The figure below shows a sinusoidal, linearly polarized wave. Since the magnitude of the H field is constrained by the E-field in a propagating wave, the rest of the discussion in this section will focus just on the E-field.

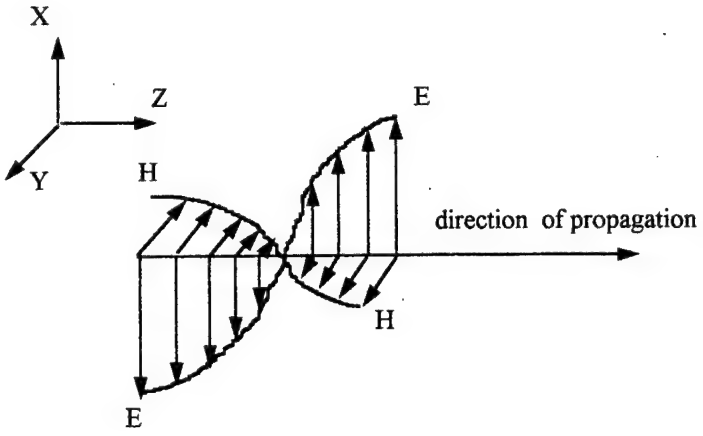


Figure 2.4. A portion of linearly-polarized electromagnetic wave.

The above figure may be misleading, however, because it only shows a very simple case of propagation, where the E-field is moving up and down in a straight, vertical line. In its general form, E can have both X and Y components:

$$\mathbf{E}(z) = (E_x\hat{x} + E_y\hat{y})e^{-j(kz-\omega t)}$$

for E(z) shown in time-harmonic form (i.e., for a single frequency $f = 2\pi\omega$.) The two components E_x and E_y do not have to have the same magnitude, nor be in phase either. If the components are in phase, and both are non-zero, the wave will be linearly polarized, but will tilt with regard to the X and Y axes, as shown in Figure 2.5(a). If the components are not in phase, this implies that the E-field components do not have to equal zero at the same time, though they do have to be sinusoidal, thus allowing elliptical polarization. The direction E-field rotates as time increases. It is called right-hand (RH) elliptical polarization if the fingers of one's right hand curl in the direction of the E-field rotation when the thumb is pointed in the direction of propagation, and left-hand (LH) if it goes the other way. Circular polarization, a special case of elliptical polarization, occurs when the E_x and E_y components are 90° out of phase, and are of equal magnitude. Actually, linear polarization is also a special case of elliptical polarization [20].

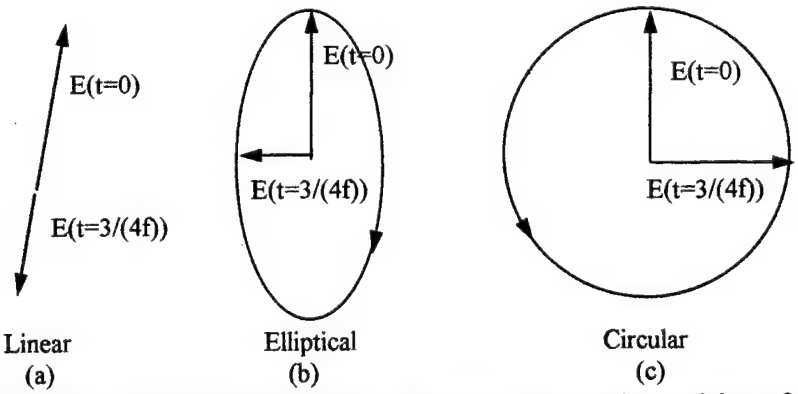


Figure 2.5. Three different polarizations, front view. If the wave is propagating straight up from the page, (b) shows left-hand elliptical polarization, and (c) shows right-hand circular polarization. The arrows in all three illustrations show the E-field at two different times: t=0, and t=3/(4f), where f= frequency (i.e., ¾ of a wave cycle after t=0).

Antennas are polarization-sensitive. An antenna that picks up only linearly polarized signals that are pointed straight up will be most efficient when receiving this vertically-polarized wave. On the other hand, a horizontally-polarized wave will not be received at all—this is what is known as cross-polarization. A linearly polarized wave that is tilted will be partially received at this antenna, in proportion to the amount of tilt in the wave's polarization. This linearly-polarized antenna will also receive only half of the energy of a wave that is circularly polarized, regardless of whether it is RH or LH circularly polarized. Similarly, a RH circularly-polarized antenna will receive half the power of a linearly polarized signal (regardless of tilt) and all the power of a wave that is RH circular, while missing a LH circular wave completely.

The expression relating the power received by an RHCP antenna versus a wave of arbitrary polarization follows: (Power received)/(Power of incoming wave) = 0.5 + (axial ratio)/((axial ratio)^2 + 1). Axial ratio refers to the ratio of the semiminor axis over the semimajor axis of elliptical polarization. It is always less than one, and is positive for RH polarization and negative for LH polarization. If a wave is linearly polarized, the axial ratio is zero and the power received is half that of the incoming signal. If the polarization is LHCP, giving an axial ratio of -1, no power is received. Full power is received only when the incoming wave is RHCP and has an axial ratio of 1. Similarly, if the antenna is LHCP, the axial ratio is negative for RHCP and positive for LHCP.

For antennas with wires in only two dimensions, polarization is constrained to be linear in the plane of the antenna. The electric field can only run in directions parallel to the wire elements. Thus, it is easy to obtain linear polarization—just keep all antenna wires in a plane. To give linear polarization in all planes, all elements must be aligned. For instance, all wires might be vertical or horizontal, as in the Yagi antenna.

Circular polarization is more difficult to achieve, since it requires a phase lag in the E-field components. Many types of circularly polarized antennas rely on phasing networks (requiring precise fabrication) to achieve this lag, significantly increasing antenna costs and sensitivity to frequency. This difficulty in designing antennas with circular polarization makes the results found with the crooked-wire genetic antenna of Chapter 6 that much more amazing.

2.2.4 Frequency dependence

The behavior of a wire antenna comprised of good conductors is dependent only on its configuration and dimensions. But it is not the absolute dimensions that determine antenna behavior, rather the antenna dimensions compared to the wavelength of the signal. Thus, large antennas that operate at low frequency can be tested with scale models using shorter-wavelength signals with accuracy, as was done for the Yagi antennas in Chapter 5. In other words, a large antenna at low frequency is electromagnetically identical to a proportionally smaller one at a proportionally higher frequency. (Recall that wavelength is inversely proportional to frequency.) There are limits to this scaling, as the resistive loss in metal is dependent on frequency, but this dependence does not make a significant difference until one is working in the tens of gigahertz.

Because antenna size, and hence performance, effectively changes with wavelength, antennas are generally at their best at a single frequency. As one deviates from this frequency, the effective size of the antenna changes, as do its electromagnetic characteristics. Bandwidth is thus defined as the frequency range over which the antenna characteristics do not change beyond acceptable limits. These limits are usually VSWR and/or gain related, e.g., a VSWR no greater than 2, or a change in gain of no more than 3dB in the main lobe.

Except for certain antennas that are designed to have the same relative dimensions at all frequencies (a situation that requires infinite size to be fully realized), all desired antenna characteristics deteriorate as the operating frequency moves further and further in either direction from the frequencies in the bandwidth. Some antennas are much more broadband than others, but all real antenna bandwidths are finite.

For many applications, bandwidth is desired to be as large as possible. It is usually given in percent, which is the ratio of the bandwidth span over the band's center frequency. For an antenna operating at 2GHz, a bandwidth of 3% would mean it would operate over a 60MHz range, from 1.97GHz to 2.03GHz. A bandwidth that is very small, less than 1%, for instance, means the antenna is very sensitive to frequency and can be used in only specialized applications. Realize also that such a narrowband antenna will be sensitive to its exact dimensions and requires exacting fabrication. A slight change in dimensions will change any antenna's optimal wavelength. Because it is so

narrowband, a slight change in this antenna's optimal wavelength may keep it from receiving or transmitting at all in the desired frequency range.

For many applications, 10% bandwidth is considered good, and 20% is considered to be very good. There are some antennas that are very broadband, though, as large as 15:1, meaning the bandwidth is 15 times the center frequency, and some are even broader still. The antennas presented in this thesis have more typical bandwidths, from just a few percent up to about 30%.

2.3 Measuring antennas

Though much can be done with computer simulation, it is of course necessary to build and test antenna designs. It is not completely straightforward to do so, however, because there are many factors that must be controlled on an antenna range to ensure data is as accurate and unbiased as possible.

One of the most important factors is the control of echoes and reflections. For this reason, antenna ranges are often built inside anechoic chambers—rooms lined with radiation-absorbing material that traps signals and keeps them from returning once they have struck a wall. Being indoors, these chambers allow complete environmental control, giving maximal repeatability and reliability.

Another important consideration for measurement is ensuring the antenna under test is in the far-field. Antennas not only generate propagating waves but also non-propagating waves that store energy and die away within a few wavelengths. These waves are large if the antenna is in supergain, but they are present even when the antenna operates normally. If one measures an antenna within a few wavelengths, these non-propagating waves will be measured along with the propagating waves. This region is known as the near-field. However, antennas are almost always used at distances well outside this near-field region, where the non-propagating waves have died out and the only signal that is received is from propagating waves. This is the far-field region. Near-field and far-field antenna performance are usually very different, therefore, when testing antennas that are intended for far-field use, one must have enough distance between the transmitting antenna and the receiving antenna to put them into the far-field from one another. One can convert near-field antenna results into a prediction of far-field performance, but the calculations are quite complicated and the measurements are sensitive to error. It is usually much simpler to just allow the few wavelengths required to put each into the far-field, though doing so can be problematic if the antennas are large and/or their operating frequencies are low.

Since the distance separating the measuring antenna from the antenna under test is important, and may need to be large, anechoic chambers are not always able to be used because they are limited in size. Outdoor ranges can be used in these cases, as they can be quite large, are generally clear of interfering obstructions, and have well-characterized environments that cause minimal disruption of the measurements. If there are covered test bays on these outdoor ranges, they are also lined with anechoic material. But even with careful calibration, outdoor antenna ranges are subject to interference from weather and returns from the ground and the environment, and thus they are generally reserved for antennas that require the large space. Besides ensuring that measurements are taken in the far-field, some antennas require tests in realistic surroundings or while mounted on full-scale aircraft that necessitate the use of an outdoor range.

The loaded monopole and the crooked-wire genetic antenna were measured in indoor anechoic chambers. The Yagi antennas were measured on a 2600-foot outdoor range. The Yagi antenna was housed in a bay 20 ft. x 20 ft. x 20 ft. lined with absorbing material good to about 800 MHz. As the Yagi antenna is difficult to measure accurately, further details of the Yagi antenna measurement equipment and procedures are contained in Chapter 5.

VSWR measurements are much simpler than antenna pattern measurements. The antenna is connected to a transmission line of known impedance and the reflections of a known signal are measured. The environment in which these measurements are taken makes little difference, so long as there is little or no interaction between the antenna and metal structures in the test area. These VSWR measurements for all antennas were made with an HP8510 Network Analyzer.

2.4 Introduction to the Numerical Electromagnetics Code, Version 2

Because the complexity in solving Maxwell's equations is tremendous for most problems of interest, numerical simulators that can solve these equations in an approximate fashion for almost arbitrary structures have been developed. If used properly, they can be quite accurate. Because affordable computers are powerful enough to solve even large problems in a reasonable amount of time, these simulators are used extensively by many antenna designers.

The numerical simulator chosen for this research was NEC2, and it was used exclusively. The NEC2 program is a very general Method-of-Moments antenna simulator. It is relatively fast, accurate and reliable for arbitrary wire structures, and uses simple, consistent text input and output files: important considerations for GA optimization where a few thousand simulations will be necessary to optimize a design, and an automated means of generating input and reading output must be used.

NEC has been around since 1981, and is in the public domain. It is easily obtained from many sources, including the World-Wide Web. Obtaining and modifying the source code is therefore simple and free of legal complications, as is copying and recompiling the simulator for different computers. It also has been used as a standard in the antenna community, and has a number of papers supporting its accuracy and describing its limitations. It has shown itself to be in very good agreement with many actual measurements, and thus there is reason to have confidence that answers received from simulation have validity. The limitations and assumptions of NEC2, along with a list of papers that have shown excellent agreement between NEC2 and measurements, is in Appendix A.

NEC2 input and output

NEC2 has a straightforward file interface for input and output. The program allows antennas to be constructed from a number of wires, which are defined by startpoints and endpoints in a 3-D coordinate system. A means of excitation—i.e., a way to input the desired electromagnetic signal—needs to be specified, as well as the desired outputs from NEC2 and any other parameters (like a ground plane) that need to be included. The basic unit in NEC2 is the wire segment. A wire segment is a portion of wire, typically between 0.1 and 0.01 wavelengths long. Each wire in the configuration can be made up of 1 or more segments, the number of which is specified in the input file. NEC2 will calculate the currents on each wire segment due to the signal source(s) using the method-of-moments. NEC2 can then determine characteristics like the far-field radiation pattern of the antenna from these currents. It is therefore possible to determine how a number of wires in an almost arbitrary configuration will behave so long as they do not violate the limitations of the simulator. A typical simulation of less than 100 segments takes less than 20 seconds to run on either a Pentium or a workstation, but scales as N^2, where N is the number of wire segments in the configuration. The compiled code used for most of the thesis limits the total number of segments to about 350. This number is more than large enough to generate antennas with characteristics of interest.

NEC2 uses ASCII text input and output files for data. The input file consists of lines called "cards" (from its early implementation when a series of punch cards was used to denote input). The figure below shows what each card does in the input file.

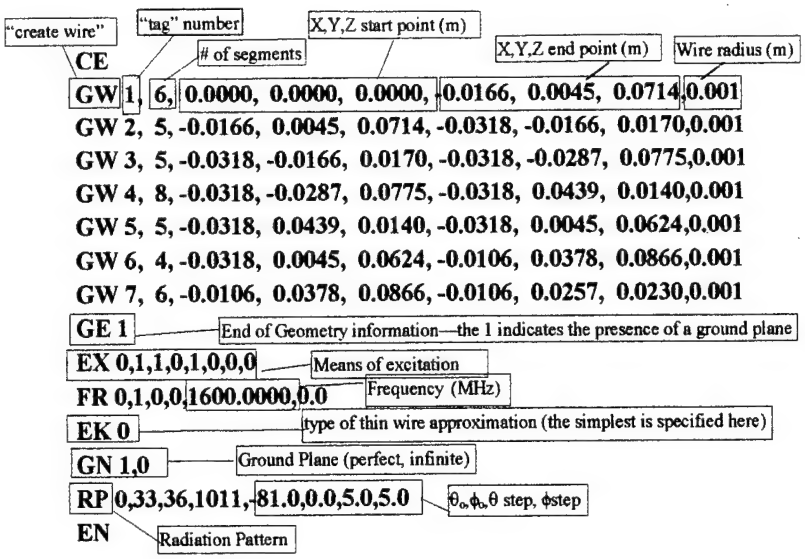


Figure 2.6. A Typical NEC2 Input File

NEC2 generates a long output file that contains all relevant simulation results, including the antenna impedance at the point of excitation and a table for the antenna radiation pattern which includes gain and polarization data for every requested angle. In order to use NEC2 as part of an objective function for optimization, then, the optimizing program must generate an input file, execute NEC2, read the relevant information from the output file and compute the objective function value based on that information.

2.10 Conclusion

This chapter has introduced the reader to various antenna concepts and characteristics that will be used in later chapters, along with a brief discussion of NEC2. Though this chapter was brief, it is hoped that the reader now has enough background to understand the concepts and terms of later chapters. For a more thorough introduction to antennas, the reader is referred to [33] and [34]. The next chapter will introduce the reader to Genetic Algorithms (GAs), the key to automating antenna design in this thesis.

Chapter 3: An Introduction to The Genetic Algorithm

3.1 Overview to the GA: an imitation of biology

The instructions for the design for each living organism are contained within the chromosomes of that organism. These instructions are coded into a base-pair sequence on long strands of DNA. A number of these strands make up the entire instruction set from which an organism is designed and maintained throughout its lifetime.

Similarly, designs in just about any engineering discipline can be reduced to a number of instructions or specifications: lengths, diameters, materials, etc. Naturally, these specifications can be arranged into a series that will completely specify a particular design. For example, take the design of a sheet of paper. Assuming that one is only going to use one basic material and process to make the paper, there are four obvious specifications: color, length, width, and weight (which, along with length and width, will determine thickness).

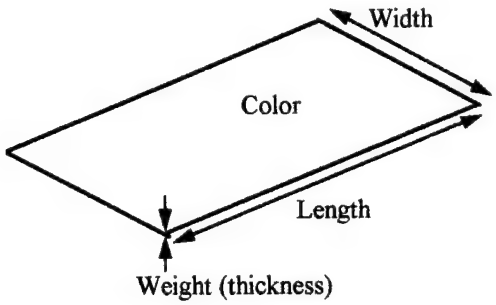


Figure 3.1. The design variables for a piece of paper

One can arrange these specifications into a string of numbers if desired, so long as a uniform coding system is established. For instance, if color 1 is designated to be white, color 2 purple, and color 3 black, that the length and width are given in inches, that the weight of the paper is given in pounds/5000 sheets, and that all these specifications will be arranged in the above order, one can completely specify a sheet of paper by the following ordered set of numbers: [1 11 8.5 20]. This particular series of instructions specify a piece of paper that is white, 11 inches long and 8.5 inches wide, and 20lbs/5000 sheets (i.e., standard thickness). Similar strings of numbers can be constructed for just about any design.

In reducing a design to a series of numbers, something remarkable has been accomplished. This series of numbers acts in a manner identical to a chromosome in an organism: it gives the information necessary to build a particular design, so long as the coding procedure is understood by the builder. But this similarity goes beyond simple cosmetics: one can mate and mutate sets of these chromosomes, and set up a procedure similar to a life cycle for organic life to optimize this chromosome and obtain an optimal design.

In the biological world, chromosomes and their resulting organisms go through a number of important processes: birth, survival-of-the-fittest, mating and the production of children, and death. In going through these processes, a species of organism adapts and optimizes itself to its environment. It is possible to simulate such processes for the chromosomes constructed for an engineering design to optimize it. Doing so is the essence of the Genetic Algorithm (GA).

The reason one might want to do such a thing is that the biological, original form of the GA has shown itself to be very powerful and robust. The environment is probabilistic and sometimes goes through radical changes, unfit individuals sometimes survive to mate, fit individuals sometimes die prematurely, competition with other species for resources is fierce and fluid, predators and diseases abound, and yet the various species survive and thrive. The

systems and processes of life are very complex and filled with an incredible number of variables, yet the GA is able to take them all into account and optimize a species for survival. It would be hoped, then, that a simulated version of the biological GA will have the same robustness and ability to optimize. As Professor John Holland discovered when he first created the GA in its current form, the computerized version of the GA does indeed have some of these properties [21]. Though the problems most engineers wish to optimize are usually static and much simpler than those for biological life, the GA can be adapted to solve a wide range of these problems effectively. In addition, there is an inherent robustness in the process that makes the process insensitive even to its own parameters. While it is important to choose reasonable values for these GA parameters such as population size, the process works very well even with parameters that are less than optimal. In fact, non-fatal bugs in GA programs are often hard to find, for though the GA may not be making the progress it could if it were bug-free, it often appears to be optimizing normally even when bugs exist.

A GA, then, is an iterative optimization process that imitates the adaptation and evolution of a single species of organism. Using a chromosomal mapping system like the one for the paper example above, the GA starts with a large number of potential design configurations. The range of possible configurations are determined by the constraints of the problem and the method of encoding all configuration information into the chromosome.

The GA process is shown in the figure below.

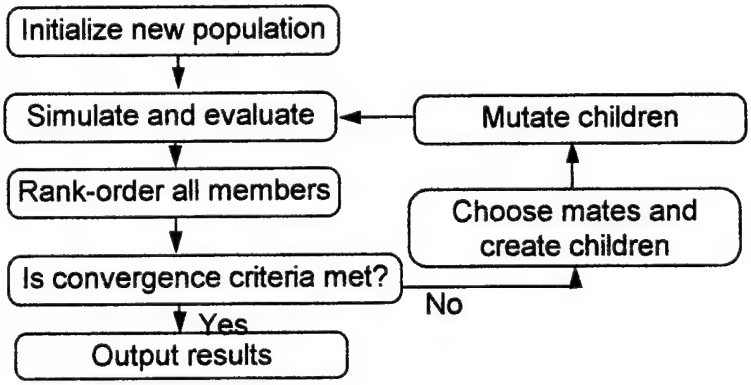


Figure 3.2. GA process flowchart

To begin the optimization, the GA selects a small set of configurations, nearly always at random. This set of configurations is called the population, just as in biology. The GA evaluates the performance of each member of the population with a cost function that compares individual performance to the desired or ideal performance and returns a single number to the GA that is a measure of its fitness. In antenna design with a GA, evaluating the cost function involves simulating each member with an electromagnetics code and comparing the results with what is desired. As in the evolutionary process of "survival of the fittest," high quality strings mate and produce offspring, while poor quality strings are removed from the population. Offspring can be created through many different procedures, each of which is essentially a method of combining information from two or more parent chromosomes to form a child with the potential of outperforming its parents. With succeeding "generations," the quality of the strings continually improves and an optimized solution is ultimately obtained. The GA method of antenna design is analogous to a method of breeding race horses, only the "horses" are antenna designs and the "race track" is a simulation to determine antenna performance. "Champions" will have many offspring, while those who do not perform well will perish without offspring. In this manner, after a few thousand simulations, a good solution is usually obtained, even when the problem to be solved has many interrelated unknowns, and a large, complex, and spiky search space. In fact, the larger the problem, the better the GA will perform over other methods of search. The GA is highly resistant to becoming trapped in local extrema, which allows it to work well for search spaces like those of the antenna design problems described in this thesis.

This feature of the GA is of great benefit to the engineer. In most engineering problems, one knows how to specify a design, but not what the best parameters are, or even an initial guess at them. Most problems of interest have many

variables, and it is very difficult if not impossible to find the optimal solution to a problem though intuition, experience, or classical optimization strategies (e.g., Newton, Quasi-Newton, and conjugate-gradient methods). These problems usually have many local extrema that can trap classical search strategies, as well as non-differentiable search spaces.

However, a GA is not only difficult to trap in local extrema, it is able to find optima when there are many variables, complex interactions, and no gradient information available. It is not even important what area of engineering the problem falls under. The GA can be and has been used to solve mechanical engineering, software engineering, artificial intelligence/artificial life, robotics, aeronautical engineering, electrostatics, as well as electromagnetics problems.

Setting up a GA, then, requires five major steps: setting up a simulator, defining the problem, determining the objective function, determining chromosome mapping and processing, and setting the GA parameters. In the next part of this chapter, the process and elements of the GA will be described in detail. This chapter will then conclude with a detailed example going through the steps to set up a GA and what occurs during the running of a GA.

Before delving into the setup of a GA, it will be helpful to cover a few terms that will be used. Beside "population" and "chromosome," several other terms are borrowed from biology. The words "genotype" and "phenotype" refer to a specific set of instructions in a particular genetic code and the resulting physical organism respectively. For instance, the specific set of numbers that designate a normal sheet of white paper given above is a genotype. The piece of paper built using those instructions is the phenotype.

While the term "gene" refers to a single element in a chromosome, "allele" refers to a specific value a gene can take. For example, in the paper example, there is a single gene that indicates the paper color. The allele set for that gene is a finite list: 1, 2, and 3. On the other hand, since the gene for length is a real number, a very large list of alleles are possible for that gene. The number of alleles possible for a gene is an important consideration that will be discussed regarding chromosome mapping and processing.

"Locus" is a term referring to the location of a particular gene in a chromosome. The paper color gene, for instance, occupies the first locus in the chromosome. The distance between loci and their position in the chromosome play a role in the effectiveness of the GA, and will be discussed.

3.1.1 Step 1: Set up the simulator

The first step in constructing a GA is setting up the simulator that will be used to evaluate all the designs generated by the GA. However, this step can be interchanged with the next step, defining the problem, depending on the situation. As described in Chapter 2, the NEC2 code was used for all antennas in this thesis. NEC2 is a very general wire antenna simulator. A simulator like NEC2 that is able to solve many different problems is very useful to explore before trying more specialized simulators. In choosing a simulator, there are many things to consider: simulation time, validity and limitations, input and output methods, and the availability of simulator and source code.

First, the time to simulate should preferably be at most in the tens of seconds. As will be shown in later chapters, a typical GA run will require a few thousand simulator runs to converge to a desirable answer. If the simulator takes longer than 1 minute to run there are several alternatives: find a faster simulator even if it is less accurate, decrease problem size, decrease number of variables so that the number of runs can be reduced, refine the simulator code or use faster computational hardware. There are also techniques that will allow the GA to use a smaller number of runs that will be discussed here and throughout the remainder of the thesis, particularly in Chapter 7.

Using a less accurate simulator is often a good solution. Recall that the GA is very robust and can operate even under noisy conditions. In most cases, a GA needs only an accurate relative ranking of individuals rather than perfect knowledge of absolute performance. Thus, a simulator that has relatively poor absolute accuracy but is able to properly distinguish between poor performers and good performers will often suffice for a GA optimization.

Problem size can also be decreased. In finite element simulations, segment size can usually be increased beyond what one would normally use, because, as stated above, the GA only needs a relative ranking of individuals to be effective.

If there are fewer variables in the problem, obviously the search space will be smaller. This will give the algorithm an increased likelihood of finding the correct solution, and decreasing in many cases the number of simulations that will be required.

If all the above fails, one may wish to refine the simulator code to streamline it for the specific problem at hand. This is usually a difficult undertaking. Alternatively, one may wish to use a faster computer or allow the simulator to solve problems across a network of machines running in parallel.

Other ways to streamline the GA itself will be discussed later. Regardless of the technique used, however, the GA will usually need at the very least a few hundred runs to converge, and this will limit the types of problems it can solve.

The next item to consider when setting up a simulator is its validity and limitations. Naturally, one must check the literature and ensure that a simulator's results correlate well with reality, otherwise one will most likely be disappointed with the results. A literature search regarding the validity of NEC2 is contained in Appendix A.

It is important to understand the limitations and underlying assumptions of a simulator. A GA with a less-than-ideal coding can easily violate the basic assumptions of the simulator and arrive at designs that simulate well but are not physically feasible. Such limitations for NEC2 are also contained in Appendix A. How to implement limitations into the GA chromosome will be discussed in Step 4.

Next, it is important to have an easy method to interface the simulator and the GA. In most high-level languages, like C++ and Pascal, it is possible to call and run programs from within a program. In this way, for instance, a GA can create an input file for a simulator like NEC2, call NEC2 to run, and then process the output file. The key is for the simulator/GA interface to require no user interaction. The simulator must be able to take input from a file or command-line parameters and output the results into a file that the GA can read and process. For this reason, graphical CAD-type simulators may be a challenge to interface to a GA.

Finally, when choosing a simulator, consider source code availability and modification difficulty. Many times one will not need to modify a simulator, but if one is having trouble with the GA/simulator interface it may become necessary. One may wish to even incorporate the simulator into the GA code itself. However, it is recommended to use an external simulator that does not need much if any modification to keep the results valid and the interface simple. Many codes like NEC2 are long (over 10,000 lines of code) and use global variables, data structures and variable names that make it difficult to include directly into the GA code. Another factor is that many simulators are in a different language from what one may use for the GA code. For instance, NEC2 is in FORTRAN, while the GA code used for this thesis is in Pascal. This fact made using an interface much more attractive.

Once a simulator is chosen, it must be configured to work with a GA. Specifically, it may need slight modification to allow automated operation. NEC2, for instance, needed very slight modification to remove prompts for the input and output filenames and instead "hardwire" certain filenames into the simulator. Memory limitations may also need to be imposed so that the GA and the simulator can run simultaneously.

3.1.2 Step 2: Define the problem

While the process of setting up the simulator will naturally limit the problems a GA can solve to a certain class, it is necessary, obviously, to define the specific problem which one wishes to solve. One must define all constraints, all unknowns and variables, and all goals of optimization.

First, one must define the constraints of the problem. It is important to include simulator limitations in the list. Determine whether they are physical, simulator or intuitive limits, and be suspicious of intuitive limits—as will be seen later in this thesis, the GA is a strategy which can explore non-intuitive search spaces. Ensure that the constraints are not mutually exclusive and are as uncorrelated as possible. Find ways to relax constraints when possible—often a GA

will find answers where one does not expect them. Incorporating constraints can be accomplished in chromosome mapping or in the objective function, as will be discussed later.

Alongside the constraints, define unknowns and variables. One must determine how many unknowns will be variables, and which will be constants. Consider memory/problem size limitations, time limitations, ability to control/modify in fabrication or use, and the sensitivity of objective to the unknown in question. Naturally, there will be constraints on unknowns and their feasible ranges which must be included in the constraint list. One must also determine whether a variable is discrete or continuous. If discrete, determine the number of different values the unknown will have, and if continuous, determine the range(s) and, if it is to be made discrete for binary encoding (discussed later), determine the number of levels it will have.

Next, one must determine the goal(s) of the optimization. List all the characteristics the GA is to optimize. Because the GA will only optimize characteristics specifically included in the objective function, one must make sure the list is exhaustive, or take chances on unlisted but desired items. List the information that is required from each simulation. Make this list as short as possible but still include all that is needed. Use experience and theoretical knowledge here. For example, in antenna patterns that cover a band of frequencies, many angles need to be simulated for a given frequency. Conveniently, increasing the number of simulated angles is inexpensive timewise. However, simulating the structure at different frequencies is expensive—each frequency point requires a whole new simulation for NEC2. Fortunately, these frequency points can be sparse, so only a small number of frequency points are simulated, yet that data is very useful.

3.1.3 Step 3: Determine the objective function

The objective function consists of a series of steps that converts the data from a simulation to a quantitative measure of performance. It can include one criterion of performance or many.

An objective function, as implemented in this research, consists of two major steps: conversion of simulation results into measures of quality, and conversion of measures of quality and constraint penalties into a single number—the fitness of the individual. This function must include all characteristics that are to be optimized.

It should be noted that it is possible to have an objective function that does not return a single number, but two or more to give a multi-criteria measure of performance. These objective functions require special ranking techniques that are significantly more complicated than single-fitness-number objective functions used in this thesis.

First, a simulator will often give output that is amenable to visualization. For example, NEC2 gives a table of angles and their associated gains and polarizations at a particular frequency. It also gives the simulated impedance of the antenna at that frequency. However, these data are not immediately useful to the GA. The GA requires a single number that gives a measure of the fitness of an individual. Therefore, for one to use NEC2 as a simulator for the GA, one must devise a way to read the numbers from NEC2 and turn them into relevant measures of quality, before the objective function can combine these measures into the fitness score.

For instance, a relevant measure of quality in antenna design is VSWR. NEC2 gives the complex impedance of an antenna, i.e., the resistance and reactance. One must therefore implement the series of mathematical expressions that convert the NEC2 data into the desired VSWR.

However, the NEC2 output of gain at various angles may be useful in its raw form, so long as one knows which angles are of interest. Often, I wanted the gain in a particular direction as a measure of quality, and then the maximum gain across a range of angles as another. Therefore a processor had to be set up that would extract the needed values from all of those returned by NEC2. Sometimes polarization loss is taken into account also, requiring the objective function to read polarization data from NEC2 and modify the gains through a series of expressions.

Measures of quality can be maximized or minimized. If a measure of quality has no identifiable ideal, then maximization is a good choice. If an ideal is easy to determine, then one may wish to simply minimize the difference

between the ideal and the individual. However, if one has more than one measure of quality, and some are to be maximized and others minimized, one must take care how they are combined into a single fitness.

Another feature that can make these measures tricky to combine is that they can behave very differently depending on how they are processed. They can be linear, exponential, or logarithmic in nature. If a particular characteristic is desired to have more importance the further from optimal it is, one may want to make it change exponentially. If an engineer does not care about the quality measure in a certain range, one can make the measure flat in that range [35].

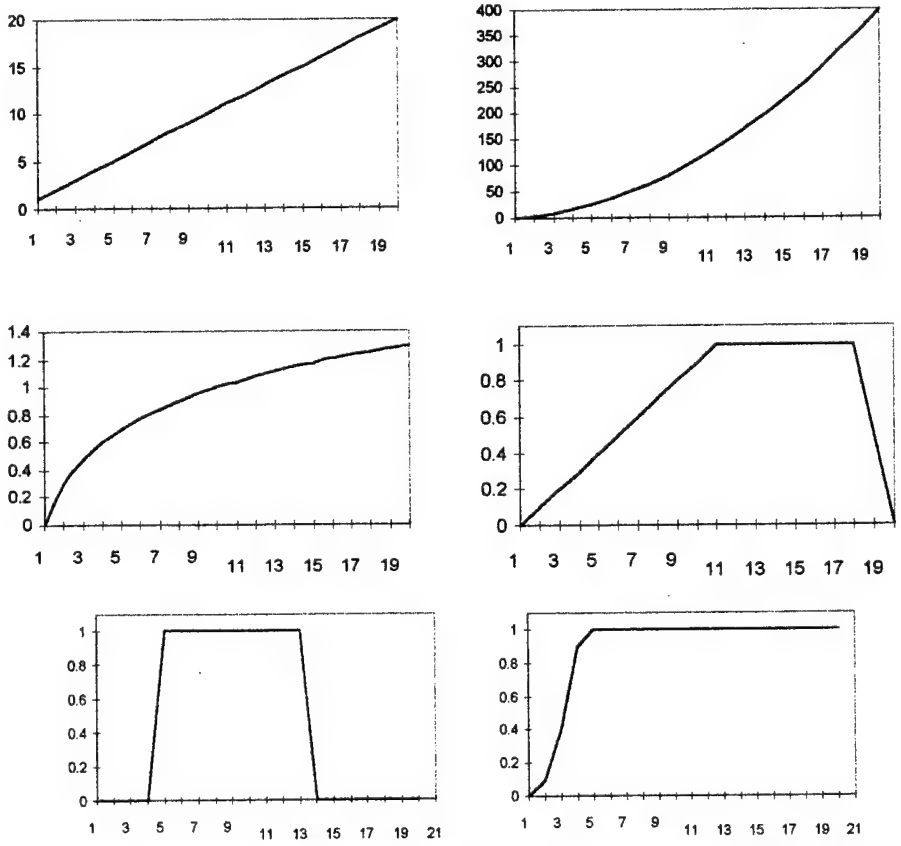


Figure 3.3. Various possible profiles for terms in a fitness function. Horizontal axes indicate the value of the measure of quality, vertical axes indicate the value of the term in the fitness function.

Once all measures of quality are determined and can be culled from the simulator data, one must create the equation to combine them all into a single measure of fitness. There are two basic methods to create such functions.

One can add the measures together modified with constants, such as in the example below.

$$\text{Fitness} = X + (c_1(Y) \cdot Y) - (c_2(Z) \cdot Z) - (c_3(T) \cdot T)$$

When one is looking to maximize the absolute value of this expression, it is resistant to annihilation, but prone to dominance. In other words, it is less sensitive to characteristics that are poor if all else is good. Under minimization to zero, it is resistant to dominance, but is slightly prone to annihilation—one bad parameter can send the score higher than its competitors. Rules of thumb for constants are: a factor of 10 between variables will produce a very great difference in emphasis. Factors of 2-3 are often good for emphasizing the most important items. Constants like c_1 and c_2 do not have to be constant, linear or even continuous. They can change with the level of the quality measure.

One can also use multiplication:

$$\text{Fitness} = (X) \cdot (c_1(Y) \cdot Y) \cdot (c_2(Z) \cdot Z) \cdot (c_3(T) \cdot T)$$

When maximizing absolute value this method is resistant to dominance, but prone to annihilation—one bad parameter reduces score tremendously. Under minimization to zero, the expression is resistant to annihilation, but prone to dominance—one good parameter near zero will cover all others.

In later chapters, the reader will note that in almost all cases of multi-goal optimization, addition was used in the fitness function. This method of combining measures of quality seems to work over a number of different situations and is relatively insensitive to the precise values of the constants used.

Though one usually wants to create a chromosome that can never violate constraints of the problem, as will be discussed in the next section, it is sometimes much simpler and nearly as effective to allow individuals to violate certain constraints and be penalized for doing so by the cost function. In these cases, it often more useful to let the objective function evaluate an individual for satisfaction of constraints before optimization objectives. It is then possible to use a pre-processing step before a simulation is performed to determine whether an individual meets all constraints. If not, a score can be assigned to the individual indicating the severity of the violation without wasting a simulation on an invalid individual.

For instance, let there be 10 resistors in series being optimized. Each can take on values from 1 to 100 ohms, but the total series resistance can be no more than 500 ohms. It would be somewhat tricky to code a chromosome that allows the full range for each resistor and ensure the total is within the constraint of 500 ohms. It may make sense to use a preprocessing step where the resistances are summed and the fitness penalized for violation, skipping any further simulation if the 500 ohm constraint is violated.

Whether or not a preprocessor of this type is used, one must devise a non-deceptive combination of objectives and constraints. In other words, an invalid individual can easily be confused with a valid poor performer if care is not taken to show a difference between them. Usually constraint violations should be penalized in such a way that clearly identifies it as a violation, not just a poor individual. Also, constraint violators should not all be given a single bad score—larger violations should be penalized more than small ones. In this way, the GA will have an indicated direction toward satisfying the constraints.

Though it is possible to handle constraints in the objective function, it is best to allow chromosome mapping to handle most, if not all, constraints automatically. It is of great benefit to have as many of the individuals that can exist through the chromosome mapping fit all constraints. This handling of constraints was done for the antennas that were optimized in this research—all possible individuals met the constraints. How to do so will be described in the next section.

3.1.4 Step 4: Determine chromosome mapping and processing

Crucial to the operation of a GA is the mapping of the design into a chromosome. There are several choices to be made while determining this mapping, and rules of thumb for making these choices will be given here.

Each chromosome in the population is a complete, independent design that specifies all variables. There are several different types of chromosomes that one can choose.

The most obvious chromosome is the 1-D chromosome like that given for the paper design example. There are many other types of chromosome mappings as well: 2-D and 3-D chromosomes, list chromosomes, and tree chromosomes, to name a few. (While this thesis will describe 1-D and tree chromosomes in detail, the reader is referred to Appendix B for a brief treatment of the other types.) Each type has its own crossover and mutation operators. Though the most common type is the 1-D chromosome, the other types bear consideration when making the decision to use one over another. It is important to choose the type of chromosome that is most native for the problem defined in Step 2, not only because it will be easier and more intuitive to code and work with, but because the GA search will be more effective with a coding that reflects the problem to be solved.

With many of these chromosome types, particularly 1-D, 2-D, 3-D chromosomes, there is still a decision to be made. Does one wish to use a chromosome with discrete alleles for its genes, or does one wish to use a chromosome with

continuous genes, i.e., real numbers? The type of chromosome that is chosen will affect the GA and its results profoundly. There are many factors that should be considered. Most important is whether the problem at hand involves continuous or discrete variables. The chromosome mapping that is able to span the search space and determine variables most naturally is probably the best choice, both for ease of setup and use, and for ease in finding a solution.

Another consideration is whether one wishes to keep to the intuitive and biological nature of the GA. Chromosomes made entirely with binary numbers are the most commonly used. There is a large body of research and theory on these kinds of chromosomes and the ways to process them. There is a close similarity between these chromosomes and their biological counterparts, in that biological chromosomes, genes usually have only two or three possible alleles. The low cardinality of the allele sets allows for mathematical treatment of the functioning of the GA, as will be shown. It also allows for an intuitive grasp of the principles involved in constructing and running a GA.

On the other hand, chromosomes that use real numbers generally need different operators to be used effectively. The allele sets for such chromosomes are, of course, virtually infinite, and thus to efficiently optimize such chromosomes requires different techniques. These techniques make use of the fact that, though there are an infinite number of alleles, they are related to one another mathematically. Because the ways to set up and process these two types of chromosomes are so different, the next two sections discuss in detail the ways to implement and process discrete and continuous-valued chromosomes respectively.

Regardless of the type of chromosome chosen, however, all variables determined in Step 2 should be formulated in such a way so that as many constraints as possible are factored into the mapping process. It is best to have all of the individuals made possible by a given chromosome mapping be feasible. The reason is that the search space for a GA consists of all possible combinations of genes. A search space with many infeasible individuals will needlessly complicate the search process, as the GA will have to sift through that many more worthless individuals to optimize the design.

There are ways to factor in such constraints into variable definitions without a lot of computational overhead. Naturally, nearly all variables have upper and lower limits, and these are easily incorporated into the mapping procedure. However, constraints that include more than one variable are more difficult to accommodate.

Constraints of this nature fall into two categories: equality constraints and inequality constraints. For example, a constraint that $X + Y = 35$ is obviously an equality constraint, whereas $X + Y < 35$ is an inequality constraint. While a chromosome mapping could simply be [X Y], where each is allowed to vary over the range 0-35, such a mapping would allow these constraints to be violated for many of the possible individuals. In the first case, a better chromosome mapping would only include one variable, say X for simplicity, which would determine Y automatically through the expression $Y = 35 - X$. In the case of the inequality constraint, the first gene would designate X, which would be allowed to vary from its lowest limit to 35, while the second gene might designate the percentage of the difference between 35 and X that comprises Y. For example, a chromosome [15 0.4] would mean that $X = 15$ and that $Y = 0.4 \times (35-X) = 8$. In this way, X and Y are forced to meet the inequality constraint without resorting to objective function penalties.

3.1.4.1 Discrete/Binary 1-D chromosomes

Chromosomes of this type have been used since Professor Holland's creation of the GA. [22] As in nature, these chromosomes have very small allele sets, almost universally limited to 1s and 0s, i.e., a binary coding. Binary coding has advantages in that the cardinality allele set is as small as possible, it is a relatively intuitive coding, it is easy to adapt to different situation, and it is easily analyzed mathematically and theoretically.

Because these chromosomes are so often binary, this section will be limited to this binary case. It should, however, be noted that it is possible to have a different discrete allele set if one desires, though the user would be encouraged to keep the cardinality small in any case. If the cardinality of the allele set becomes large, the GA will need a very large population to have enough combinations of alleles to effective process the search space, or the mating and mutation operators will need to use relationships between the alleles, as in the case of real-numbered chromosomes.

Chromosome mapping

The mapping of a given design into a binary chromosome would proceed much like the paper example above, but instead of real numbers representing the various values, different binary sequences would represent variables. For instance, the standard sheet of paper design [1 11 8.5 20] could become [01 10110 10000 10011] under the following rules: the first two bits indicate the paper color, allowing a total of four different colors to be specified, three if 00 is undefined or redefined as another of the already-designated colors—the latter is recommended so that the coding does not contain undefined individuals. The next five bits are used to designate the length of the paper, in increments of 0.5 inches from 0.5 at 00000 to 16 inches at 11111. Thus, 10101, when converted to base-10, is 21, which indicates a length of 11 inches. Similarly, the next five bits indicate the width of 16 half-inches plus the minimum 0.5 inches, or 8.5 inches. The last five digits indicate the weight of the paper, given in pounds from 1 to 32 pounds. 20 pounds is indicated by 10011 (19 pounds plus the minimum 1). Note that though a different resolution or range for the variables could have been determined, the number of levels for the last three variables was limited to 32 by the use of five bits. If more levels were desired, the number of bits used would have to increase. But the use of more bits increases the length of the chromosome and likewise the size of the search space. An increase in search space size increases the difficulty of searching it, so the tradeoff between having necessary resolution versus size of search space must be considered. A rule of thumb is to have the minimum amount of resolution possible that will be sufficient to include an optimal answer, or at least an indication of an optimal answer that can be explored after the GA finishes. If the resolution is too low, the optimal solution will be likely to be left out of the search space. The number of bits used for a single real number should reflect whether a variable is sensitive or robust, and constraints on fabrication.

As can be seen from the paper example, using a number of bits to quantize continuous variables, both continuous and discrete variables can be accommodated by binary chromosomes.

Mating operators

The mating operators involve the processing of parent chromosomes to form children. Each different operator has different advantages and disadvantages, and the right choice depends on the problem and its search space. These mating operators are all varieties of crossover—a process of swapping genetic material modeled after crossover in cellular biology. Because this process in the GA is an imitation of biology, and some of the resulting effects of crossover are similar to those in biology, it is useful at this point to discuss the biological process of crossover.

In organisms, crossover occurs during the meiosis process which forms the gametes (the sex cells from which children are created). The vast majority of all organisms have two complete sets of chromosomes, one from each parent. During a process called meiosis, all the chromosomes in the organism are replicated. Then the chromosomes from one parent are paired up with their homologs. (A homolog is a chromosome from the other parent that matches a given chromosome.) It contains the same genes (genes that determine eye color, blood type, etc.), though perhaps with different alleles from its matching chromosome. As each chromosome is already duplicated, this matching up of homologs produces sets of four chromosomes. These foursomes are lined up across the cell and then pulled apart, and through the course of two cell divisions, each strand will eventually be in its own gamete cell separate from the other three from its foursome [23].

However, something one might not expect occurs during the separation process. As the chromosomes are separated, strands will often touch at least at one point. Where the strands touch, they split and rejoin with the other, so that two new strands are formed: one that is part strand A and the other part strand B, and a second strand that is the exact opposite. When the separation occurs, these new strands become new chromosomes that behave and are treated just like the original, uncut strands. However, the new strands have unique sets of alleles that are unlike the combinations the parents contributed [23].

The most basic operator for the GA is single-point crossover. This is very similar to the crossover that occurs in organisms. A single crossover point is chosen at random between any two bits in a chromosome. The front of the first parent's chromosome is combined with the second part of the second parent's chromosome to form a new chromosome. This process is shown in the following figure.

Parent chromosomes: [00011|110]
{11001|100}

With a crossover point between the 5th and 6th bits, as shown above
Offspring: [00011]{100}

Figure 3.4. Single point crossover

There has been some controversy about whether a chromosome should be allowed to have crossover points in the middle of a series of bits that forms a single functional gene, like a single real number that is distributed over a number of bits. It appears that the best choice is to allow such crossover points for a few reasons. First, if one does not allow crossover between such genes, one has increased the cardinality of the allele set. If no crossover can occur, the gene is a single indivisible unit in the chromosome, and that implies that its cardinality has to be the same as the number of possible values for that gene—if a 5-bit gene, it will be a cardinality of 32. As mentioned above, there is a significant disadvantage to high cardinality without special operators. In addition, in biological organisms, crossover can occur between functional genes as well. Crossover does not occur simply on gene boundaries but between any two sets of base-pairs on the DNA strands. Hence, there is also biological precedent for the allowance of crossover points between any two bits in the chromosome.

Another biological concept, linkage, needs to be discussed. In organisms, most characteristics independently assort. To explain this concept, let an organism have two alleles for each of two different genes, a and A for the first gene, and b and B for the second gene. The four possible combinations of these alleles in the gametes are: (a, b), (A, b), (a, B), and (A, B). If these genes are on separate chromosomes, they independently assort, and each of these combinations will occur with equal frequency [23].

It has been found that there is often a link between two characteristics—the number of gametes with two of the combinations, for instance (A, B) and (a, b), is much greater than the other two. This implies that the genes are on the same chromosome, and that one of the parents of the organism must have had (A, B) in its gamete cell, while the other had (a, b). The smaller the distance between the loci of the two genes, the greater the linkage between them. This effect is due to the crossover operation that occurs during the creation of gametes from which offspring are created. If the distance is small, the genes will rarely have a crossover point occur between them, so the allele pair will pass uncut to the gametes, and then to the offspring of the organism. [23]

With single-point crossover, this linkage is seen in the GA as well. Since there is only one chromosome in this design, the only way for the genes to independently assort is if each gene were allowed to cross over one by one. Under any type of crossover with fewer crossover points than genes, then, it is important to place more highly correlated variables next to one another. There is greater linkage between variables that are close together, because, as in biology, there is a smaller likelihood that a crossover point will occur between genes that are close together than between genes separated by a larger distance. In many problems, unknowns are correlated, that is, the setting of one unknown affects the effect of another unknown. If genes of correlated unknowns are close together, it is more likely that successful gene combinations will appear in the children intact from its parents rather than split and mixed with the genes of the other parent.

The easiest modification one can make to single-point crossover is to increase the number of crossover points, to create two-point crossover. The two crossover points are chosen, as before, with the exception that the points must be different, and that they can be placed just beyond the end or before the front of the chromosome (these two positions are, in effect, the same regarding the effect of crossover). The child is formed from the front of the first parent's chromosome, up to the first crossover point. The middle part of the child is created from the second parent's chromosome between the two crossover points, and the remainder of the child comes from the last part of the first parent's chromosome, as shown in the figure below.

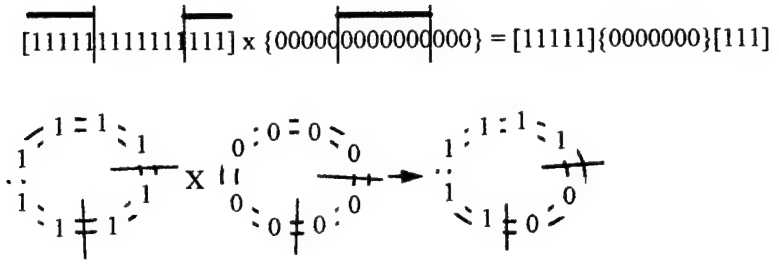


Figure 3.5. Two-point crossover, with equivalent ring-chromosome representation.

There is an advantage to having the extra crossover point: it removes the head-to-tail bias. For instance, a gene near the front of the chromosome will always be weakly linked to a gene near the other end under single point crossover. In fact, the genes actually at the ends will always be divided on every crossover by necessity. However, with two-point crossover, the chromosome can be thought of as a ring, joined at the ends. The crossover operation selects some material from one parent and the remainder from the other, but there is no longer a bias about which genes will be linked based on where the gene lies along the length of the chromosome.

One can select more than two crossover points, but studies have shown there is little value in doing so. [22] Therefore, the two crossover methods above are the major ones for the binary 1-D chromosome.

Mutation operators

Mutation is a randomization of the child's chromosome. It is important to have in limited amounts, because it ensures that alleles that are eliminated from the population are not necessarily gone forever. It allows the GA to explore combinations that do not currently exist in the genepool of the population. However, because it is random exploration, it must be performed with a small probability or the information processed with the GA will be destroyed. The amount of mutation that is useful will be discussed in Step 5.

Essentially there is only one mutation operator possible for the binary chromosome. Once the bit to be mutated is selected, the operator performs a bit-flip: changing a 1 to a 0 or vice versa depending on the current value of the bit.

3.1.4.2 Real-numbered chromosomes

As the name implies, real-numbered or real-valued chromosomes consist of a string of real numbers. There are several advantages to using real-numbered chromosomes. First, each variable can usually be encoded in one gene. The real number can be accurate to several decimal places, and the number of levels such a gene can take on is infinite for practical purposes. It even makes the design easy to read from the chromosome, in that each variable is naturally in base-10. However, the major disadvantage is that special mating and mutation operators need to be employed to handle the infinite cardinality of the allele set. But even this disadvantage has an advantage: it allows problems that are naturally real-number problems to be exploited both by mathematical techniques that use interpolation or extrapolation and the GA search technique at the same time. This is a powerful combination that can solve very difficult problems effectively. For this reason, it is suggested that one use the real chromosome for problems involving mostly real, continuous variables.

Chromosome mapping

The mapping for this chromosome is very simple. Generally, each variable requires only one gene. The paper example [1 11 8.5 20] is a real-numbered chromosome. Note, however, that the first number specifies a discrete variable. The handling such a variable with a real chromosome will be explained later in this section. However, the design of a sheet of paper is mostly comprised of numbers that are continuous. If one were to have some reason to optimize this design, it would be most natural to put it into a real-numbered chromosome. On the other hand, if the numbers were not actually continuous—for instance, if paper sizes were only available in half-inch increments—it would be more appropriate to use a binary chromosome.

If there are a few variables that are discrete, one can still use a real chromosome representation for them, and simply discretize the decoding process. For instance, a real gene can be rounded to the nearest whole number to indicate the value of a discrete variable. This processing has been done, and will be discussed in later chapters. However, the mating and mutation operators use the relationship between alleles in the real numbering system, and thus this may be less effective for discrete variables that have no such relationship. But the GA optimization process is so powerful and robust that even such inefficient methods will still result in effective optimizations, so long as there are not too many such variables and they are relatively minor in effect. If not, one may wish to have a real chromosome and a binary chromosome in each individual, mapping each variable into the chromosome that is most natural. Note that each will need a different set of operators.

Mating operators

Unlike the operators for the binary chromosome, the most effective operators for the real chromosome operate gene-by-gene. In this way, one can think of each variable occupying a separate chromosome, and each independently assorts with the others. It is possible to use single or multi-point crossover operators, as with the binary chromosome, but as has been stated the infinite cardinality of the allele sets become a problem.

The best mating operator I have found is actually a process involving the combination of three separate mating operations on a chromosome. It has been described and examined in [24], and will be referred to as Adewuya's method. The steps for this method follow:

1. Choose the three parents
The following steps are performed for each gene separately:
2. Perform Quadratic crossover
3. If 2 fails, due to lack of the desired extremum in the feasible gene range, perform heuristic crossover with the best and the worst parents of the three
4. If 3 fails to produce a child gene in the feasible range, randomly choose one of the three parents to have its gene copied into the child's gene.

The next two sections define what is meant by Quadratic and Heuristic crossover.

Quadratic crossover

Quadratic crossover is a method of predicting the best gene value for a child based on a second-order fit to the gene vs. fitness value for three selected parents. Though this method will sometimes misleading the GA in the creation of a child, more often than not it should be able to generate better children. The figure below shows where a child would be chosen should fitness be desired to be minimized and the three parents shown are chosen to mate.

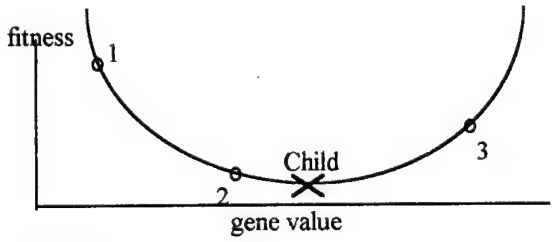


Figure 3.6. Quadratic crossover example.

Following is the mathematical procedure for quadratic crossover.

Let there be three parents:

$$P1 = \{p_{11}, p_{12}, \ldots, p_{1n}\} \text{ with fitness } f_1$$
$$P2 = \{p_{21}, p_{22}, \ldots, p_{2n}\} \text{ with fitness } f_2$$
$$P3 = \{p_{31}, p_{32}, \ldots, p_{3n}\} \text{ with fitness } f_3$$

The quadratic curve for each gene in the chromosome is given by:

$$h_j(x) = a_j x^2 + b_j x + c_j$$

so one must determine a_j, b_j, and c_j.

The analytic expressions for the three constants are:

$$a_j = \frac{1}{(p_{3j} - p_{2j})} \left[\frac{(f_3 - f_2)}{(p_{3j} - p_{1j})} - \frac{(f_2 - f_1)}{(p_{2j} - p_{1j})} \right]$$

$$b_j = \frac{(f_2 - f_1)}{(p_{2j} - p_{1j})} - a_j (p_{2j} + p_{1j})$$

$$c_j = f_1 - a_j p_{1j}^2 - b_j p_{1j}$$

The critical point (extremum) of h(x) is found by taking the first derivative,
$d\, h_j(x)/ dx = b_j + 2\, a_j\, x = 0 \rightarrow x = - b_j / (2a_j)$.

If $d^2 h(x) / dx^2 = 2a_j < 0$ (if maximum is sought) or > 0 (if minimum is sought), then $x = - b_j / (2a_j)$.

Let the child $D = \{d_1, d_2, \ldots, d_n\}$. Then the child's genes are given by $d_j = - b_j / (2a_j)$,
if the second derivative is properly negative or positive, and d_j is within the limits of feasible genes.

Heuristic crossover

Heuristic crossover is a method of extrapolating the value of a child's gene that is likely to be more fit than its parents based on a line drawn between the two parents. An example is shown in the figure below. The distance of the child's gene from the parents' is random but limited by the distance separating the two parents.

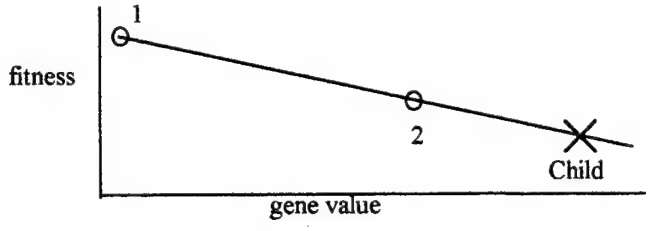


Figure 3.7. Heuristic Crossover example.

The mathematical procedure for heuristic crossover follows.

Let there be two parents:

$$P1 = \{p_{11}, p_{12}, \ldots, p_{1n}\}$$

$$P2 = \{p_{21}, p_{22}, \ldots, p_{2n}\}$$

Let P2 have a better fitness level than P1, and let the child $C = \{c_1, c_2, \ldots, c_n\}$. Then the child's genes are given by:

$c_i = rnd\, (p_{2i} - p_{1i}) + p_{2i}$
where $0 < rnd < 1$.

There are other methods contained in [25] which were not used in the course of this thesis, and which have been shown to work less effectively by [24] than the above method on many problems, but may still be of interest to the reader. They are listed in Appendix C.

Mutation operators

There are many more mutation operators possible with a real chromosome than with a binary one. A few are presented here, though the most commonly applied in this thesis is the Gaussian mutation method. Most involve manipulating the probability distribution of the mutation.

The first mutation method draws a mutated gene from a uniform probability distribution between the upper and lower limits, regardless of the original value of the gene. This is a simple method, but will usually place the mutated gene far from the original. Since an entire variable is usually contained in a single gene, this can wreak havoc on a chromosome's fitness, as one of the variables has just been randomized. This kind of randomization infrequently happens with a binary chromosome, where a bit flip can affect a multi-bit variable by moving it only halfway across its range at most.

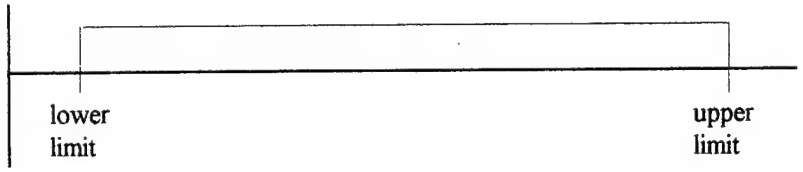


Figure 3.8. Uniform probability mutation.

Gaussian mutation is similar to uniform probability mutation, except that the mutated gene is drawn from a Gaussian distribution centered around the original gene's value. This allows mutated genes far from the original to be rare, allowing exploration to be concentrated around already-successful values. A standard deviation for the distribution must be set as a parameter. This deviation can be decreased as the run progresses. This mutation operator pulls the mutated gene from a Gaussian distribution centered around the original gene. I generally gave this distribution a standard deviation of 10% of the gene range (thus restricting the mutated gene to be within +/- 30% of the gene range of the original gene 99% of the time).

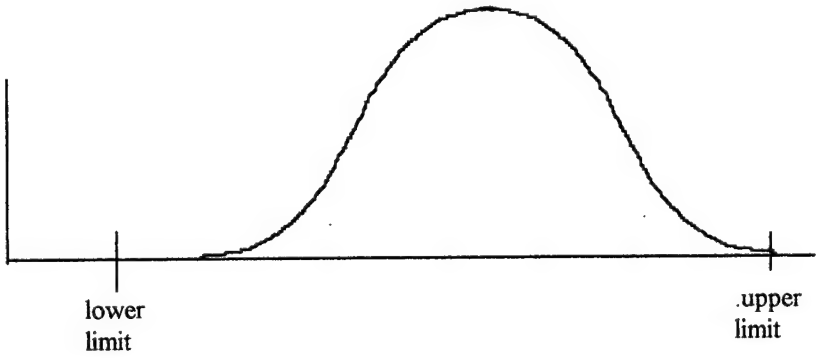


Figure 3.9. Gaussian mutation

Boundary mutation pegs the gene to one or other of its range limits. This is a simple mutation that can be used in situations where the edges of the range have special significance, or where it is desired to shake up the population more than usual. It can also help with interpolation methods like quadratic crossover.

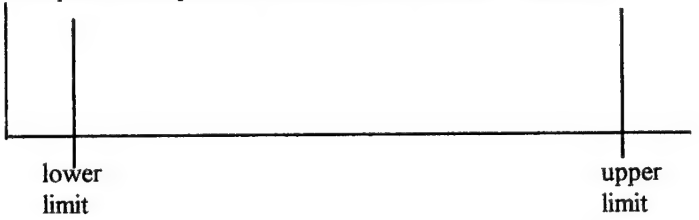


Figure 3.10. Boundary mutation

Non-Uniform mutation is a procedure where the amount of mutation grows smaller as the GA run progresses. As will be discussed later, the GA generally converges to a small range of solutions as it reaches its conclusion, and it makes sense to limit the amount of mutation as the amount of variation in the population decreases.

The mathematical form of non-uniform mutation follows:

$C_i = \{C_i + \Delta(\text{gen \#}, \text{upper limit} - C_i)\}$ or $\{C_i - \Delta(\text{gen \#}, C_i - \text{lower limit})\}$

where $\Delta(\text{gen \#}, y) = y \cdot \text{rnd} \cdot (1 - \text{gen\#}/(\text{maximum gens}))^b$, $0 < \text{rnd} < 1$, b is a parameter, C_i is the ith gene of the child chromosome. Whether the gene is mutated upwards or downwards is determined by a virtual coin-flip. This expression allows less and less dramatic mutation as the run progresses.

3.1.4.3 Tree chromosomes

Tree chromosomes are so named because they resemble trees in their structure. They have nodes containing information and branches connecting them. The nodes can contain either data or instructions. The branches show connections or paths. Tree chromosomes are essentially fractal structures.

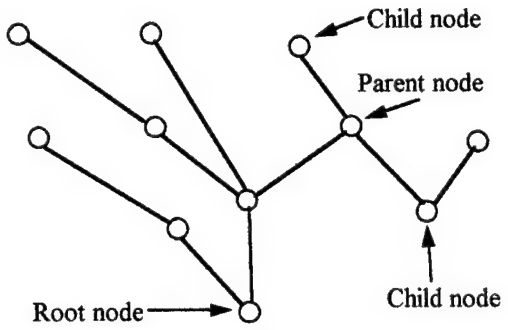


Figure 3.11. Tree chromosome

For example, tree chromosomes are often used to generate programs using a GA. The nodes contain logical expressions like "not," "and" and "or," or data to be manipulated. The programs are evaluated and evolve like other types of solutions.

In the field of antenna design, however, the tree chromosome has a different use. The nodes contain spatial data—the coordinates of an absolute location in space or a distance and direction that can be used to determine a point relative to another point. The branches are used to show connections between points. In this way, an antenna can be constructed from the tree chart and the data in the nodes.

It is important to note that there are two terms used when discussing tree chromosomes that are also used in GA terminology that mean different things: parent and child. In tree chromosomes, as shown in the above figure, a node is designated a parent when it is connected to at least one node further from the root of the tree. The node or nodes it is connected to are called the children of that node. This is not to be confused with parent and child chromosomes, which refer to entire individuals.

Chromosome mapping

As mentioned above, for antennas these chromosomes are used to determine connections and placement of wires. The information in the nodes contain placement data, and the branches indicate connections.

The data in the nodes can be expressed in two different ways: in absolute coordinates, or in relative coordinates. With absolute coordinates, the data in each node consists of three coordinates—either X,Y and Z or R, θ, and ϕ

depending on the desired coordinate system. These coordinates are used as points in space between which the different branches are connected.

When using relative coordinates, it is possible to use either two or three coordinates in the nodes. The absolute location of the endpoints of a wire are determined by the absolute location of the node's parent. One end is located at the parent's coordinates, and the other is located at the parent's location plus the coordinates contained in the node in question. If one wishes to limit the length of the wire to a constant, then one only needs two coordinates such as θ and ϕ to determine a wire. In this way, an antenna can be comprised of many identically-sized wire segments.

Mating operators

Due to the fractal nature of the tree chromosome, the mating operators are different from 1-D cases and have different effects though the same basic function of mixing data from parents to produce children. There are two common operators: subtree swapping and node swapping.

Node swapping can occur between any two nodes in the two parent trees. A child would be essentially identical to the first parent with a node from the other parent. This is a relatively minor change, and so is usually an auxiliary method to the major method of subtree swapping.

Subtree swapping involves swapping parts of two trees to create a child. These two parts do not have to be at all similar. See the figure below for an example.

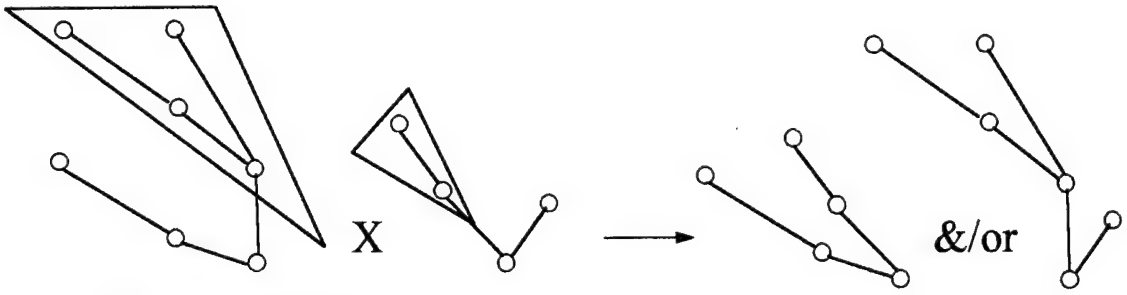


Figure 3.12. Subtree swapping crossover

Note that the tree can actually grow larger using this system. It is possible for an entire tree to be added to a node in another tree. This is a radical change from the other chromosomes presented thus far. They are all restricted to a number of genes set from the beginning. The tree chromosome can become very large with a large number of generations. It is usually wise to place an upper limit on the trees or a penalty for rising above a particular size to keep simulations reasonable.

Mutation operators

There are four mutation operators in common use for tree chromosomes. Two of the mutation operators—subtree swapping and node swapping—are identical to the mating operations, though there is only one tree involved. The other two operations are destruction of subtree and node replacement.

Subtree swapping and node swapping forms of mutation occur between any two nodes/subtrees in a single individual just as it does in the mating operation. Subtree destruction, as shown in the figure below, simply involves removing a subtree from any node in the tree. A restriction should be placed on this mutator so that if only one branch comes from the root, it cannot be chosen, as this would make a null tree. Node replacement uses one of the mutation methods described in the real chromosome section to replace the original contents of the node with new data.

Two example antennas designed using tree chromosomes are briefly described in Chapter 7.

3.1.4.4 The mating and selection process

Regardless of the chromosome mapping used, a means of selecting which chromosomes are carried over from the previous population and which chromosome mates with which must be established.

There are two major GA types, and their difference lies in the way they execute the mating process and the process by which the fittest are chosen for survival (the two processes are related). These types are the simple GA and the steady-state GA. They each affect the "choose mates and create children" and "mutate children" blocks in the GA flow chart (repeated below). All other functions are the same.

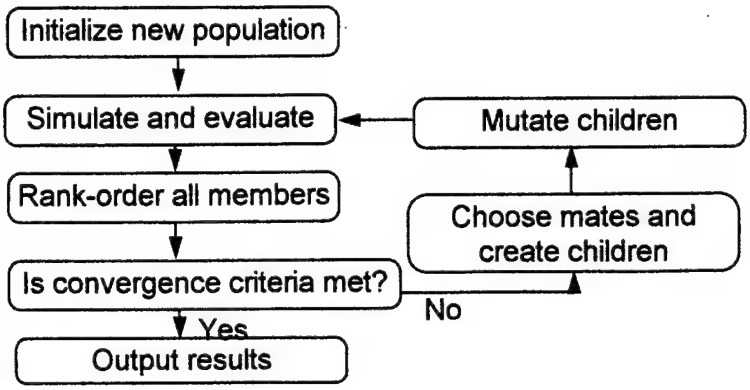


Figure 3.13. GA process flowchart

The simple GA is discussed at length in [22]. It is a fairly close approximation of nature in the way chromosomes are chosen to survive and reproduce. In essence, the simple GA replaces the whole population each generation with new children, though some are copies of parents.

There is a variation of the simple GA called the elitist simple GA. In the elitist GA, the best member from a generation is kept to the next generation under all conditions. Though this breaks with nature, one does not usually want the GA to lose its best solution. It may take a very long time to rediscover such an excellent solution, if it finds one at all. Therefore, it is usually advised to use the elitist variation of the simple GA.

The simple GA is composed of three major operations: reproduction, crossover and mutation. In the first stage, reproduction, chromosomes from one generation are selected to carry over to the next. No new chromosomes are created yet, but some individuals from the old population will be left out, while others will have multiple copies in the new population. Whether a chromosome has a good chance of being reproduced depends on its fitness level: the higher the fitness in relation to its peers, the higher the chances it will survive.

The next step is crossover. With some designated probability, members of the reproduced population are crossed with their neighbors in the manners described above and which the user has determined. Usually the probability of crossover is high, perhaps around 80-90%. More will be said about this parameter in Step 5.

Mutation then occurs in this new population with some small probability, usually less than 1%. More will be said about values of this probability later.

The advantage of the simple GA is that it is simpler to analyze, simpler to understand, and closer to natural genetics. However, because the population is renewed each time, it takes much longer to converge. For problems involving time-consuming simulations, this can be a serious problem.

On the other hand, the steady-state GA automatically keeps a fraction of the population from generation to generation. This fraction, made up of the top performers from a particular generation, can range from 10% to over 90%. The children can either be generated from the entire population before removing poor performers, or they can

be generated from only the top percentage. In the latter case, poor performers are simply replaced by children, and have no offspring themselves.

Because this process keeps the top performers, it is more focused on exploitation over exploration. Even if the entire population is used to create children, it converges much faster than the simple GA. If only the top fraction are used as parents, it will converge significantly faster still. However, it is less like nature (individuals from prior generations are kept as potential mates), and it may converge too quickly, that is, there may be so much exploitation there is too little exploration to solve the problem most effectively.

In this thesis, I used the steady-state GA where only the kept percentage were allowed to be the parents for the next generation—the method with the fastest convergence. This was important, in that the simulations were costly in terms of time, as is the case with most electromagnetics problems.

Regardless of whether the GA is simple or steady-state, it must have a method of selecting chromosomes in proportion to a figure of merit. There are a couple of methods used, namely the tournament method and the weighted roulette wheel method. Either one can be used with either style of GA.

The tournament method randomly selects a small group of chromosomes from a population. The most fit member is chosen from this group to be a parent. Another set is chosen and the most fit member of that set is taken as the second parent. Alternatively, both parents can be chosen as the two (or three) most fit members of the first random group.

The most common method, however, and the one used exclusively in this thesis, is the weighted roulette wheel.

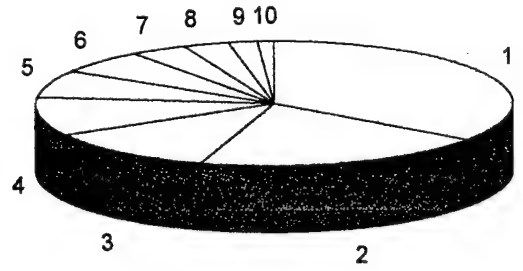


Figure 3.14. The weighted roulette wheel.

In this method, the roulette wheel is split between the various candidate chromosomes. It is split unevenly, though, giving a larger share to those chromosomes that are more deserving according to certain criteria. By far the most common weighting is done according to fitness, but the wheel can be weighted by other characteristics like similarity to a particular chromosome.

In order to set up a wheel in the computer, a few steps can be taken. The first step is optional but should be considered. One may wish to convert fitness scores for a more equitable weighting. Let an illustration show why one might want to do so. As a population converges, the scores of the individuals will become similar. They may all be clustered around an average, say 100. The best individual may be 100.1, while the worst may be 99.9. With scores so close, the roulette wheel will be evenly split between individuals. However, it may be that the 0.2 difference between the top and the bottom score is significant to the engineer. In that case, one may wish to rescale the scores by subtracting the lowest score so that they range from 0 to 0.2, which will show a large difference in the wheel weighting.

As was discussed earlier, one can either minimize or maximize the fitness function, however, if one has chosen to minimize, one will need to transform the scores by subtracting all scores from the worst parent, so that the best one ends up with the largest score, and the worst has the smallest. In this way, the wheel can be apportioned properly.

This also automatically scales the scores as was described above. Since all optimizations in this thesis were minimizations, this rescaling was done routinely in all GA optimizations.

Once scores have been transformed, all scores need to be added to get the total fitness. In symbolic form, Σf is desired, where the summation is over all potential parents.

From Σf, the proportion of the wheel to give the ith chromosome is given by the expression: $f_i / \Sigma f$. One then apportions slots to the parents.

A common way to approximate the wheel is with a large 1-D matrix, where chromosomes are assigned the number of slots that corresponds with their designated fraction. For example, say there are seven parents in a population, each with decreasing fitness. The 1-D matrix could look like the one below.

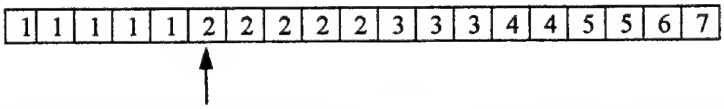


Figure 3.15. 1-D matrix that approximates the weighted roulette wheel. The arrow corresponds to the result of a sample "spin."

The problem with this method is that it has inexact probabilities. The number of slots a chromosome fills rarely corresponds exactly to what it should be given. There are different methods for apportioning remainders, but none will make the matrix proportions exactly right.

Another, more accurate method is to apportion a continuous range to each parent, and to make each parent's range adjacent to the others'. A random number is generated, and the range in which its value falls indicates which parent was selected by that "spin." An example of such a wheel is shown below.

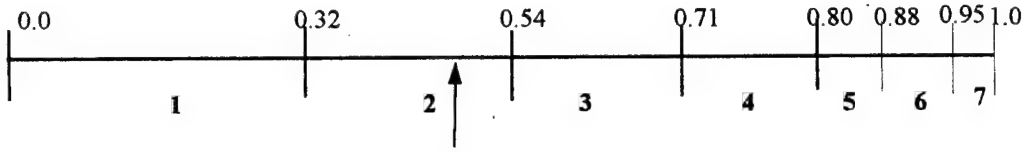


Figure 3.16. Continuous range method for determining roulette wheel ranges. The arrow corresponds to the result of a sample "spin."

3.1.5 Step 5: Determine genetic algorithm parameters

There are two parameters common to both the simple GA and the steady-state GA: population size and probability of mutation, In addition, the simple GA has probability of crossover, while the steady-state GA has percentage overlap as a parameter. While the GA is very robust to these parameters, it is still important to set them in the ballpark of their optimal setting. Certain of these parameters are more sensitive than others.

As will be shown in the example later in this chapter, and in other chapters, the most important parameter is population size. Population size determines the size of the genepool, which will determine the amount of the search space that is searched during the run. It may seem intuitive to use the largest population possible, but that is not always the optimal case.

Populations that are too large can cause problems. They move slowly in response to good points found, and they have many children in each generation, thus requiring many simulations. They will often be more consistent in the solutions they arrive at, but they are not always the best answers. The reason is that a large population can overpower the genetic material from an above-average individual that resides on a narrow spike in the search space. The large population and its genepool diversity overpowers these individuals by sheer numbers, and generally it is the larger hills that will be exploited. If larger hills do not contain the best answers, the answer will be often be suboptimal. In addition, it takes many more simulations to arrive at good answers with a large populations, as will be shown. It may make sense, then, to attack a problem with several runs of a small-population GA rather than with just one run of a large-population GA as both will often cost the same in terms of simulation time.

On the other hand, populations that are too small generally do not have enough diversity for the GA to search much of the space. They converge quickly to suboptimal points. It is important, then, that one choose a good population size for the problem.

Those who work with GAs on other problems have used anywhere from 10,000 members to only 30—thus the correct number is often a result of the specific problem to be solved, the GA style (simple or steady-state) and the chromosome mapping and processing.

In the process of doing the research in real chromosomes, it seemed that a population between 100 and 200 individuals worked well for the antenna problems solved. It does not seem to be affected by the number of genes that are used—chromosomes have ranged from 6 genes to 44 genes, and they all have been optimized effectively by a population of 175. This robustness of population size to chromosome length is not a complete surprise, given the infinite cardinality and the methods of crossover for real chromosomes. Unlike binary chromosomes, the genes of the real chromosomes are all mated separately, so each one must have sufficient diversity in the genepool, but it is not as critical to have many different combinations of allele values.

The next most important parameter is probability of crossover for the simple GA and the fraction of the population saved from generation to generation (i.e., the overlap) for the steady-state GA. Though a simple GA was not used in this thesis, others have found that probability of crossover should be 80% or higher. [22]

For the steady-state GA, 30% overlap from generation to generation seems to work well in many different cases. (30% overlap implies that 70% of the population is replaced in each generation.) The GA is somewhat more robust to this parameter than to population size, but the GA runs faster and better with 30% overlap from run to run than with higher values. Lower values seem to restrict the amount of exploration too greatly.

For either style of GA, the probability of mutation should be around 5% or less. 0.6% was commonly found to produce good results. This probability is the chance that a particular gene in a chromosome will be mutated. Though this percentage of mutation could be implemented by going gene by gene in each child, and doing the equivalent of rolling a die to see if it would be mutated or not, in most of the optimizations here, it was ensured that a certain number of mutations occurred each generation by mutating the number of genes given by the probability of mutation multiplied by the number of genes in the new children.

Another set of parameters involves determining the point at which the GA is halted. There are many possible convergence criteria that can be used to stop the GA. A very simple one is to stop the GA after a given number of generations. Other criteria are: stopping the GA after a given number of simulations, after a certain period of non-improvement (e.g., after 5 generations where the top score does not improve), after the diversity of the genepool drops below a certain threshold, or when the user inputs a stop signal. All of these have been used in the antenna problems discussed in future chapters.

Stopping the GA after a number of generations is a very arbitrary method. The GA may or may not be truly done optimizing a design at 70 or 90 generations depending on the GA parameters. It is important to not stop the GA too soon. In addition, it could be that the GA has been done for some time and has been wasting time getting to the set number of generations. Usually there are other criteria that show doneness better than a number of generations.

However, sometimes the GA cannot be allowed to run beyond a certain length of time, because an answer is required or because the resources are not available. Thus, one may wish to designate a certain number of simulations after which the GA quits. This may seem identical to stopping the GA after a given number of generations, but it is not. Particularly with a binary GA, some individuals are cloned in the genepool as the run converges. One can set up a routine that searches for such clones, and simply copy the score from the original chromosome. There is no need to resimulate them. The computational overhead of a GA is generally so much smaller than that of the simulations that the time the GA takes to optimize is a function primarily of the number of simulations that are performed. Thus it is often useful to designate a number of simulations as the stopping criteria.

The GA usually starts off improving rapidly, and then the amount of improvement generation to generation decreases until it virtually stops. A way to tell when the GA is finished, then, is to monitor the amount of improvement over the course of several generations. If very little or none is seen then one can fairly safely assume the GA is finished. However, depending on the problem being solved, there may be movement in the average of the genepool score while the top individual is not making progress. For example, a GA may discover an excellent individual that is far superior to the next best individual. In this case, this top individual is likely to be too dissimilar from the rest of the population for a mating to produce a child better than itself. Thus, there may be no improvement in this top individual for some time while the rest of the population more slowly catches up to the top individual. Once the population has closed the gap and enough similarity exists between the top individual and the rest, improvement in this top individual can be made. A variation of this criteria, then, is to watch the average of the population or some upper fraction of the population. When it stops moving, chances are the whole GA has finished its optimization. Usually one will want to give the GA several generations to show improvement, especially if the problem is tricky to optimize. It may take a few generations in some cases to show improvement. Again, one does not want to stop the GA too soon.

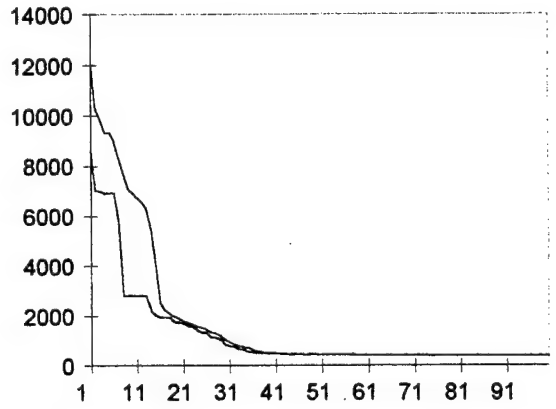


Figure 3.17. Best fitness, Average fitness vs. generation. The optimization was working towards minimization. The lower line is the best score, the upper line is the average of the parent scores.

When the diversity in the genepool of a GA becomes very small, the power of the GA is diminished and it becomes a type of stochastic hillclimber, and it should be stopped. Usually, the GA has stopped making useful progress long before the genepool converges that much.

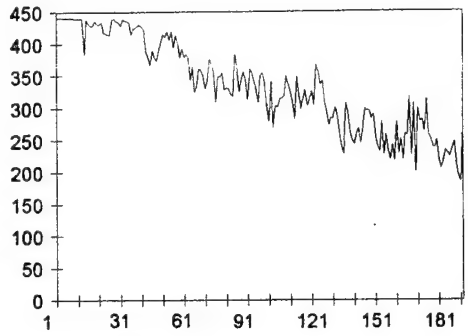


Figure 3.18. Diversity in the population in a sample run with real chromosomes. Notice how it has a downward trend, but is very noisy.

Often, the user has various reasons for wishing to stop a GA run: time allotted has expired, no progress is being made, or the GA seems to be converging to an undesirable answer, so it is useful to allow the user to stop the run when desired. It is then necessary to have the GA output data to the screen so the user can track its performance. This leads to the next section: analyzing GA performance, during and after a GA run.

3.2 Analyzing GA performance

As a GA runs and after it has completed, it is possible to determine how well it is doing. This can be important, because the GA procedure is so robust that it may still run even when there are bugs in the code, but it will not perform well. It is good to be able to monitor the progress of the GA rather than set it running and see how it did days later, especially if the code has not been well validated. Cost functions may not be quite what one needs to get the desired results, constants and variable constraints may need tweaking, and bugs may need to be removed.

It is also important to have data saved at the end of the run that can be used to analyze performance, in case one did not watch the run in real-time, or if one wants to see the progress of the GA over time all at once. Monitoring the GA and saving useful data will be discussed in the next two sections.

3.2.1 Monitoring a GA during a run

Several parameters are useful to have posted periodically as a run progresses. These characteristics can be split into two groups: individual qualities, and population-wide qualities.

Certain qualities are desirable to be visible at all times. They are: generation number, member number currently under simulation, size of member in terms of simulation time required, and whether a member is identical to another member already simulated and hence is not going to be simulated. After the simulation is completed or the score is copied from another member, it is useful for the GA to show certain critical measures of quality and the score of member under test.

At the end of each generation, it is good to compile certain statistics and print them at the end of the generation or after every simulation to ensure the numbers are always on the screen alongside the member data listed above. These are: the best fitness and worst fitness in the parent population, the number of generations the average parent score has remained constant, the number of generations the best chromosome score has remained constant, and the diversity in the parent population. This diversity is shown to help the user determine if the GA is close to convergence. If the chromosomes are binary, one measure of diversity is the total number of bits of all the parents that are different from the best chromosome. If the chromosomes are real, the range of each gene in the parent population can be used to indicate diversity. Note that the real chromosome diversity data in this case is not a single number, but a series of numbers, one for each gene in the chromosome.

A couple numbers are good to print after each generation only: the genotype of the best chromosome, and the number of simulations run so far. During the debugging process, though, it is necessary to monitor much more when

running and checking for bugs. Particularly prone to invisible bugs are the mating and mutation processes, which will still allow the GA to run, but will hinder its optimizations.

3.2.2 Evaluating GA performance after a run

After a GA run has completed, it is useful to have certain records that show the progression of the run. It is important to output all GA parameters: population size, percent saved, mutation rate, number of crossover points, etc. One must also, of course, output the best chromosome and it is good to produce its phenotype in the form of its input file to the simulator. It is also useful to record the total number of simulations needed to get to this answer.

One should set up a brief "fossil" record that records during each run. Items to be included are: best chromosome and score of each generation, average score of each generation, diversity at each generation, and cumulative number of simulations at the end of each generation.

Sometimes one will want to halt a run, then pick it up again later. If one uses a steady-state GA, using only the top fraction of the population for parents, one can save the entire run by recording the above parameters and the parent population's chromosomes and scores at the end of each generation. To restore the run, then, one need only load the genepool with the last generation's parents' chromosomes and scores and begin the generation process. Keeping this kind of detailed fossil record is very useful, especially as insurance against power outages or computer glitches during particularly long runs.

Following is a table that may aid the user in troubleshooting a GA that is not working well.

If...	Try...
It converges before getting to a good solution	Increasing exploration: increase the population, steady-state GA—increase percentage overlap (simple GA—lower crossover percentage), increase mutation.
It takes too long to converge	Decreasing exploration: do the opposite of above.
It goes to very different answers each time	Using speciation and niching to increase exploration of different hills. (discussed in next section)
The answer is somewhat poorer than you anticipated, especially from previous related GA runs	Looking for bugs! Check that you have changed over all constants and parameters if optimizing a new problem with a previously-functioning GA.
The following rows further breakdown the one above...	
If the fossil record shows abnormal improvement curves—e.g., improvement is constant and slow, even at first	Looking for bugs in the mating and mutation procedure. Insure the constants used are reasonable. Try using different population sizes, overlaps, and even crossover/mutation operators.
If the best individual shows some good characteristics and some poor ones	Checking the objective function—ensure proper scoring. Adjust constants, or adjust linearity of the measure of quality (e.g., change a linear function to an exponential).
The fossil record shows many individuals that are violating constraints	Remapping the chromosome to allow most, if not all, possible individuals to meet constraints.
If all the above has been tried and the answer is still not optimal	Reorder the genes in the chromosome. Try using different variables or expressions for variables, e.g., instead of using X, use log X or 1/X. Combine variables in natural ways, e.g., if a problem is sensitive to a ratio X/Y, use X/Y and Y as a variables instead of X and Y. Try to make variables as robust as possible, with large valid ranges for the GA to work with.

Table 3.1. Troubleshooting table for common GA problems.

The GA is a powerful optimization strategy, but it will work better in easier search spaces than in difficult ones. Since one's goal is to find the best answer, one should think through the best way to set up the GA to ensure the best

results are obtained. It is not so powerful that one does not need to take any care in specifying a problem. One should use their knowledge of the problem to encode the chromosome, cost function and GA operators in the most easily solved, most native manner.

However, there are still more tricks that can be employed to enhance a GA's ability to optimize a problem efficiently. The next section will discuss these enhancements.

3.3 GA modifications to increase efficiency

It is usually important to ensure one is getting the most from simulations run. The GA in its raw form can be inefficient when it comes to function evaluations, especially the simple GA. Designs are often large and take a long time to simulate, so inefficiency may make the problem intractable, and one may want to get to the answer quickly, especially if it converges to a wrong answer so one can run it again. There are two ways to enhance a GA's efficiency: use techniques that increase its efficiency directly, or use techniques that enhance the exploitation aspect of a GA, thus allowing it to converge to an answer faster. Both methods will be discussed in detail.

3.3.1 Direct efficiency enhancements

There are several ways to enhance efficiency in a GA. Those that will be discussed are: eliminate redundant simulations, minimize problem size for the simulator, employ curve fitting techniques, begin with coarse searches that lead to refined searches.

As alluded to previously, with binary and other discrete chromosomes, individuals will begin to clone themselves as the GA converges, especially near the end of the run. It is therefore useful to institute a large file or array that stores all simulated chromosomes and their scores. At the least, an array that contains the parent chromosomes and their scores should be maintained. This array is then searched before a new chromosome is simulated, ensuring that redundant simulations are eliminated. Unfortunately, this technique is limited when using real chromosomes, for little or no exact repetition occurs then.

One can increase efficiency by minimizing the problem size for the simulator. One can use coarse simulations that take a fraction of the time a real one does. So long as there is a strong correlation between the coarse simulation and a more accurate simulation, i.e., an individual that scores high relative to others with a coarse simulation would also score high relative to others with a fine simulation, then the coarse simulation can be used by the GA. Then the accurate simulation can be carried out on the best individual to determine its true characteristics. For instance, one may use only two frequency points in a band to determine an individual's frequency characteristics, thus saving valuable simulator time, and then evaluate more carefully once the GA has completed. This can be done if the antenna has a relatively predictable frequency response based on the answer at these points.

The coarseness can be incorporated into the amount of data received, as in the above case for frequency dependence, or it can be incorporated into the number of elements simulated in a finite element simulator like NEC2. During this research, it was advantageous to use a number of segments that was low—low to the point of being close to violating NEC2 model assumptions, without actually doing so. A few percent of accuracy was sacrificed, while decreasing the simulation time considerably. The careful validation of GA results after a run showed this approach to be valid—performance was rarely significantly worse under careful scrutiny.

Another method is to use proper convergence criteria, stopping the GA before it begins to make only minimal progress. One can experiment with different criteria that indicate that the GA has completed most of its forward progress. This method can be effectively coupled with another technique that will be discussed in the next section: adding a hillclimber at the end of the GA run.

One can also increase efficiency by re-using simulations as much as possible—though NEC2 did not have this capability, some simulators do not require a complete resimulation if only certain changes are made to an individual.

One may wish to minimize the number of complete simulations performed by grouping individuals to take advantage of this capability, or one may wish to limit the resolution of the variables that cause completely new simulations to be performed. One can also place a penalty in the cost function for variables that are modified that require complete simulations, to discourage the GA from trying those variables before trying the easier ones.

A relatively uncommon technique is to fit a curve to the data from the population's simulations. A GA can use multilinear regression to get a surface that can be used to predict children's scores. The GA can base simulations on predictions, using them as pre-processors so the GA only simulates high-payoff candidates. Alternatively, the GA can use the predictions to determine the parents for the next generation, though it is recommended that the GA simulate the new parents before mating, if possible. This technique is investigated in more detail in Chapter 7.

Finally, one can increase efficiency by doing multiple GA searches, starting with a very coarse search space and ending with a refined one. This is not to be confused with a coarse simulation—in this approach, one limits the resolution on the chromosome, limiting severely the number of possible individuals in a population. For instance, one can take a 105-bit, 21 unknown problem, and, using only 2 bits per variable instead of 5, convert it to a 42-bit problem. This is much easier for the GA to search. Once an answer is obtained, the GA can search only a small part of the original space, using the same number of bits or less, but with a tighter restriction on the variables. Besides allowing the GA to search a smaller space, the user will be able to quit GAs that are running up the wrong hill. Since searches will proceed faster, the user will be able to stop GAs that are showing signs that it is focusing on the wrong part of the search space, and can try again before many simulator runs are wasted on a failed approach. This technique is explored in detail in Chapter 7.

3.3.2 Indirect efficiency enhancements by increasing exploitation

A way to indirectly increase efficiency is to increase exploitation in a GA. By increasing exploitation, one pushes the GA to explore the hills in the search space it has already found, rather than to look for new, perhaps better, ones. There is an obvious risk in that by limiting exploration: one may miss an important answer. However, if simulations are costly, one must increase exploitation to allow convergence in a reasonable span of time. Even with these tricks, a reasonable problem will still require a few thousand simulations. However, almost any other optimization technique will require the same number or more, unless the problem is quite simple!

One can increase exploitation by setting the GA parameters properly, minimizing the problem size for the GA, adding a hillclimber to the GA for use after convergence, institute sharing and speciation, or allow a variable population size coupled with speciation.

To increase exploitation, there are several GA parameters that can be changed. The most sensitive is population size—if the population is decreased, the amount of diversity will be lessened in proportion, and the number of parents will be decreased, and thus a good individual will be more important in the mating process and have less competition. This will ensure a good individual will have a larger proportion of the mating roulette wheel, and thus allow it to mate more frequently. As this individual will produce more children, its genetic material will spread more rapidly through the population. This will increase exploitation.

A similar method that can be used with the steady-state GA is to decrease the number of chromosomes that are used as parents while keeping the population constant. As with decreasing population size, this method allows good individuals to have less competition and mate more frequently, increasing exploitation. However, as the population is not similarly decreased, there will be an increase in the exploration of the parents' genetic combinations. More children will be produced to refill the population each generation. This extra exploration may be helpful, particularly in difficult problems where it is unlikely that a child will be better than its parents.

Another way to increase exploitation is to decrease mutation, since mutation is a means of random exploration, and is not beneficial for exploitation. One can also decrease the chromosome length—i.e., decrease number and/or resolution of variables. Remapping the chromosome to place related variables as close together as possible can be effective too, as well as using two-point crossover to eliminate head-to-tail bias. Finally, one can use binary instead of real numbers to decrease the number of simulations necessary before convergence.

Once convergence criteria are met, the GA run can be completed with a hillclimber. This has been found to be most effective with a binary GA, as a real-valued GA that has been run to completion will already have zeroed in on the parameters that produce the best results, due to its arbitrary accuracy. However, binary GAs, while good hill-finders, are inefficient hillclimbers when compared with deterministic, classical methods like quasi-Newton methods or the like. Many of these methods do not require gradient information, and though they are easily trapped in local minima, the GA has hopefully given a good enough starting point that the optimum will lie close to this initial input into the hillclimber. This technique is also described in Chapter 7 in more detail.

If speciation techniques are used, one can allow population size to grow as species develop, and/or decrease the sharing penalty. Various criteria for larger populations can be used, like an increase of modality of fitness scores, or the appearance of loci on the search space that seems to attract population members. However, this technique requires an explanation of sharing, niching and speciation, which follows in the next section.

3.3.3 Sharing, niching and speciation

In organic life, there are many, many species. A species is differentiated from others by the characteristic that it cannot mate with other species without producing defective or unviable children. Besides obvious mating difficulties, most often this is caused by chromosomal incompatibility between different species: a frog's chromosomes are very different in number and length than a cat's or a bird's.

In the animal kingdom, of course, there is advantage to speciation: different species can live using different resources, and they each comprise a separate evolving population. If a disease wipes out one species, chances are it will not harm others because of the incompatibilities. If a species cannot adapt to a changing environment, other species will take its place. This diversity allows life to survive even under times of radical change.

Species occupy specific niches in the food chain. Niches are resource-related, in that a niche is a stable place in the food web. A species must find a niche to be successful in the long term. Each species uses resources that are limited, so overpopulation is self-correcting.

However, most problems in the engineering world are static, and so the questions arise: why speciate such a static environment, and how is it done for a GA operating in such an environment? The main reason to speciate in the GA environment is to provide a way for the population to explore more than one hill at once. In a spiky search space, there are many comparable hills to climb, and each may seem equally promising at the outset of the GA search. However, the GA converges onto one of these hills during its run. Which hill is eventually climbed is determined by how easy it is to find in the search space (i.e., how much of the search space it occupies), how high it goes compared to other hills, and random chance from the population initialization and the production of children. Speciation increases exploration without requiring a larger population or an increase in random search.

Speciation, in the GA environment, is the development of different species that exploit different niches in a search space, established in a GA by fitness incentives and mating and replacement strategies. Speciation is a method of ensuring the population does not get prematurely trapped into exploiting a single hill that may be less than the optimal, before exploring other hills first.

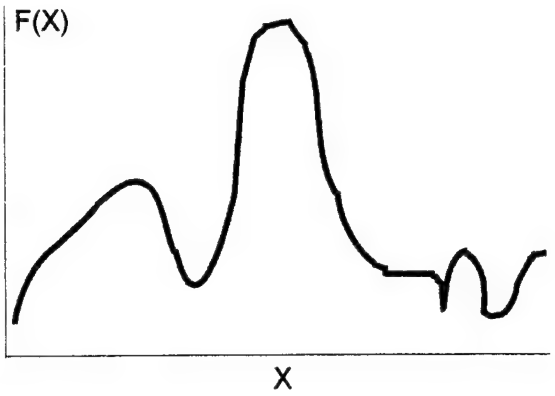


Figure 3.19. Multi-modal search space.

In a GA environment, however, there is usually no chromosomal incompatibility to preventing mating. Therefore, incentives must be given in the mating selection process and the fitness function to encourage speciation. There are several ways to give such incentives: sharing, mating restriction, and selective replacement.

Sharing, or crowding, means scaling fitness to reflect the surrounding density of population members, reducing fitness proportionally. In a sharing situation, niches are formed by peaks in the search space. As the members of the population converge on one of these peaks, the fitness of each is scaled to reflect the crowding of the population around this peak, simulating the depletion of resources that would occur if all animals required the same food source. In this manner, individuals that find other, less populated peaks will gain in fitness relative to the main cluster of the population, and their genetic material will have a chance to spread, even if it has not achieved a high position on the other hill. In this way, niches form in the population, and multiple hills are exploited without requiring a great deal more exploration.

The problem with this method is that distances between individuals must be determined. For binary chromosomes, it is possible to count the number of bits that are the same as the member in question to determine this distance. For real chromosomes, one can calculate the distance that other individuals are from the member in question using a sum of squares. The difference between the ith gene of the first chromosome and the ith gene of the second chromosome is squared and these squared distances are then summed. The square root of this sum gives the Euclidean distance between the chromosomes. In equation form,

$$\text{Distance(Chromosome D from Chromosome C)} = \sqrt{\Sigma_j(\text{Gene j of D- Gene j of C})^2}$$

Mating restriction is another method of introducing species into a GA. If one is using the weighted roulette wheel, for instance, the selection is still done with the roulette wheel, but once one parent is chosen based on fitness, the other parent(s) are chosen based on similarity rather than fitness. For instance, let parent 1 be chosen using fitness-weighted roulette wheel. The roulette wheel would then be re-weighted, giving those chromosomes most similar to parent 1 the largest share of the wheel. Distance is calculated as in the sharing method, with chromosome C being the first parent chosen by the fitness-weighted roulette wheel. However, recall that the closer one chromosome is to the first parent chosen by fitness, the larger the share of the roulette wheel, implying that distance will need to be converted so that those with the smallest distance have the largest similarity measure.

In this manner, species arise without modification required to the fitness function. Either this method or the sharing method will produce sub-populations that will evolve more-or-less independent of one another. This mating restriction method is preferred, however, because it takes into account something that occurs in nature but is ignored by the sharing method: the problem of unviable children that result from interspecies mating.

As sub-populations drift apart, they become specialized to exploiting the niche or hill they are on. When chromosomes from different hills attempt to produce children, however, their offspring will likely fall in the middle

region between the two hills. This usually results in fatalities: children with scores so low, they are not able to survive. [22] In addition, "birds of a feather flock together." Usually animals do not attempt to mate with species other than their own. The qualities that attract mates: calls, colors, etc. do not appear to a large extent in other species. If they did, the species members would expend considerable effort making offspring that could not hope to survive, and the waste of resources would probably lead to death to the species, or to the survival of more discriminating members.

In a GA that uses sharing, however, this mating restriction is not present. Only resources are rationed. While the GA is so powerful a technique that this oversight still allows good results, a mating restriction is helpful in conserving simulator runs, applying them to children more likely to survive.

The final method of speciation is selective replacement. Children replace the most similar members of the population if they are better. Often the most similar members will be one of the child's parents, especially if the chromosome is binary.

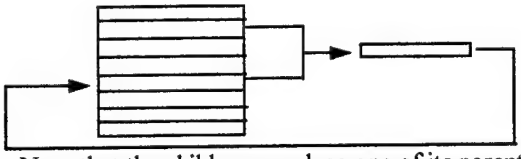


Figure 3.20. Selective replacement. Note that the child may replace one of its parents.

This also will produce speciation, in that genetic diversity tends to be preserved, yet only the better chromosome is kept. Thus, the population evolves and explores hills, yet it does not lose its genetic diversity as quickly. However, this method still faces the same problem of unrestricted mating that can cause a high proportion of fatalities in the children.

3.4 The GA Process: an example

Now that the setup and running of a GA has been described in great detail, it is useful to put all the steps together in an example. To facilitate better understanding of the GA as applied to electromagnetics problems, a simple antenna problem will be used for this example. This example is being discussed for two reasons: first, it is important for the reader to have an intuitive grasp of what happens in a GA, both in the setup and in the running. Second, problems solved with the GA in later chapters are more complex and their search spaces are difficult to visualize, so a very simple example will helpful. This problem is a simple two-element antenna.

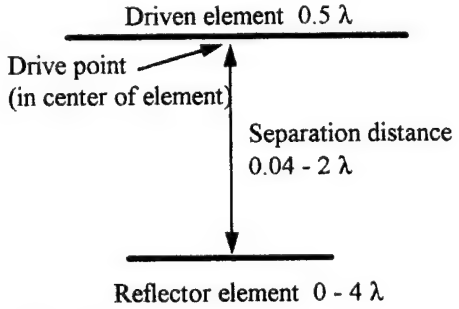


Figure 3.21. Two-element antenna search space

The front element is driven by a single two-wire source. The back element is a parasitic that is used as a reflector to increase directionality in the forward direction (straight up on the figure). In this example, it is desired to produce the maximum gain in the forward broadside direction. There are several possible variables: length of the driven element, length of the reflector element, separation distance between the two elements, offset and tilt between elements, placement of the drive point along the driven element, and wire radii. So that one can visualize the search space, the

search space will consist of only two variables: the size of the reflector and the separation distance. Therefore, the driven element will be constrained to be 0.5λ, both elements will be symmetric and aligned, the wire radii will be 0.001λ, and the drive point will be in the center of the driven element.

The response surface can be graphed easily. The response surface shows the relationship between the gain of the antenna versus the two variables in the optimization.

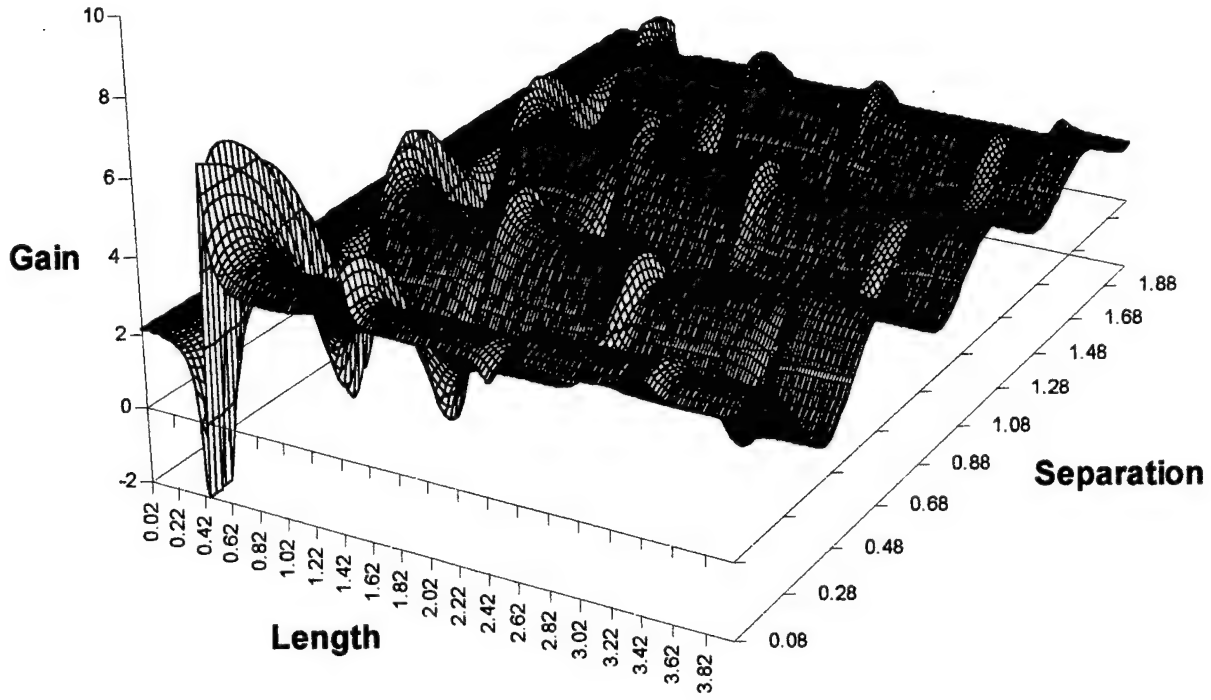


Figure 3.22. Response surface for gain vs. separation and reflector length.

The optimal settings for this antenna as shown in the graph above are 0.14λ separation and 0.48λ reflector element length. Following is a polar plot of the pattern of the best individual from the above plot. Its maximum gain is 6.9 dBi.

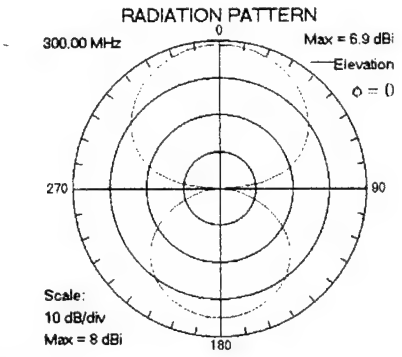


Figure 3.23. Radiation pattern of best antenna.

Now that the example problem, its search space, and its response surface have been explained, the steps to set up a GA to solve the problem will be followed. The steps are repeated here:

Step 1: Set up the simulator
Step 2: Define the problem
Step 3: Determine the objective function
Step 4: Determine chromosome mapping and processing
Step 5: Determine genetic algorithm parameters

Step 1 is simple to finish in this case—NEC2 has been chosen as capable and fast. A file transfer system is in place to move data between the GA and NEC2. NEC2 simulations take only a few seconds for this antenna, and NEC2 can handle all of the valid individuals in the search space.

Step 2 has been done in large part above. It has been determined that it is most desired to increase the gain, and ignore all other data. The variables and their constraints have been determined. A fair amount of accuracy is needed, though, in the length and spacing of the elements, from prior experience with these types of structures, so if the GA uses discrete chromosomes, it will need increments no larger than 0.02λ to have the necessary resolution (0.02λ, incidentally, is the resolution of the above chart of the search space).

Step 3 follows easily from Step 2 in this case. Since there is only one measure of quality, the GA has only to score each individual based on its gain. Hence, the fitness function will be the following:
fitness = gain at broadside (0° ϕ, 0° θ). This fitness will be maximized.

Step 4 is a bit more tricky. It will be done two ways so the reader has a chance to see both a binary and a real-valued GA at work. In the binary case, the GA will use 7 bits per variable (giving 0.015λ increments in spacing and length). Though there is an option of scrambling the genes of the binary chromosome, which will have to be 14 genes long, it will be constructed in the most typical way. The binary chromosome will be [L1 L2 L3 L4 L5 L6 L7 S1 S2 S3 S4 S5 S6 S7], where the Ls indicate the locations of each bit in the binary representation of the first variable, length, while the Ss indicate the locations of the seven bits that make up the second variable, separation.

In the real-valued chromosome, there are only two genes. Its representation is [L1 S1], where L1 is the length gene, and S1 is the separation gene.

Step 5 is dependent on Step 4, but since the problem is easy, the populations will not need to be particularly large for the GA to show normal progress. For the first example, a binary GA will be run with only 20 members, with a 2.5% mutation rate and 50% overlap from generation to generation. The GA will be stopped when the top 50% have the same bits, meaning that there is no diversity in the parent population.

3.4.1 The binary GA

An actual GA has been run with the above parameters, and will be discussed in detail. While this does not generally occur with these parameters, the GA found the global optimum. After going through this run in detail, along with the real GA optimization of this example, an experiment designed to uncover the best parameters for this optimization will be discussed.

This first generation was generated at random. Recall that scores are just the broadside gain of the structure designated by the genotype.

Generation # 1
#1 10010010000011 sc: 5.400000
#2 01101110001001 sc: 5.390000
#3 10000100101001 sc: 4.790000
#4 10011111110000 sc: 4.510000
#5 10011111110001 sc: 4.380000
#6 00011011010110 sc: 4.230000
#7 10001111100110 sc: 4.050000
#8 00001100010110 sc: 3.890000

#9 00000001101101 sc: 3.850000
#10 00000000010000 sc: 3.850000
#11 00111011100010 sc: 3.540000
#12 00110001000001 sc: 3.430000
#13 00101111011111 sc: 3.370000
#14 10001100010100 sc: 3.340000
#15 11001100110110 sc: 3.320000
#16 11110001111101 sc: 3.320000
#17 01011000111011 sc: 3.310000
#18 01001100111100 sc: 3.220000
#19 10100110111001 sc: 3.020000
#20 01101010011000 sc: 2.590000
top 50% avg score: 4.43400

To provide a pictorial way to understand the diversity and makeup of the population, histograms will be shown that depict the number of 1s for each gene in every generation. (Naturally, all genes that are not 1s are by necessity 0s.) For the first generation, the histogram is the following:

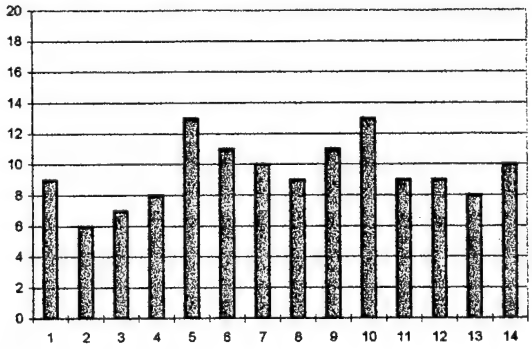


Figure 3.24. Histogram for 1s in the first generation.

Notice that the number of 1s is fairly evenly distributed. The second generation was generated, and is listed next:

Generation # 2
#1 10010010000011 sc: 5.400000
#2 01101110001001 sc: 5.390000
#3 10000100101001 sc: 4.790000
#4 10011111110000 sc: 4.510000
#5 10011111110001 sc: 4.380000
#6 00011011010110 sc: 4.230000
#7 10001111100110 sc: 4.050000
#8 00001100010110 sc: 3.890000
#9 00000001101101 sc: 3.850000
#10 00000000010000 sc: 3.850000
#11 00011111010000 sc: 4.920000 from 6 & 4, xo: 2
#12 00010010000011 sc: 3.530000 from 8 & 1, xo: 2
#13 10010110000000 sc: 5.390000 from 1 & 10, xo: 12
#14 00011011001001 sc: 3.240000 from 6 & 2, xo: 9
#15 00011011000011 sc: 3.680000 from 6 & 1, xo: 9
#16 00011011100101 sc: 3.650000 from 6 & 9, xo: 9
#17 01101100001000 sc: 5.470000 from 2 & 10, xo: 11
#18 00011010001001 sc: 2.090000 from 6 & 2, xo: 6
#19 01101110001001 sc: 5.390000 from 2 & 1, xo: 14
#20 10011111110001 sc: 4.380000 from 4 & 5, xo: 11

mutations: [16,11] [16,9] [16,2] [16,2] [13,6] [11,9] [17,7]
top 50% avg score: 5.00200

Notice that the first half of the population of Generation 2 is simply the best 10 chromosomes from Generation 1. The second half are new children, and their parents and crossover points are listed after their genotypes and scores. The mutations that occurred in the children are then listed in ordered pair form: the first number is the child mutated, the second is the number of the gene that was flipped, with the number 1 gene being the gene farthest to the left. Though highly unlikely, the 16 chromosome was mutated 4 times in a row! Notice the top 50% average has jumped up to 5.00 from 4.43, and the top score (Chromosome #17 in this case) is 5.47, as opposed to 5.40 in Generation 1. The histogram for this generation is:

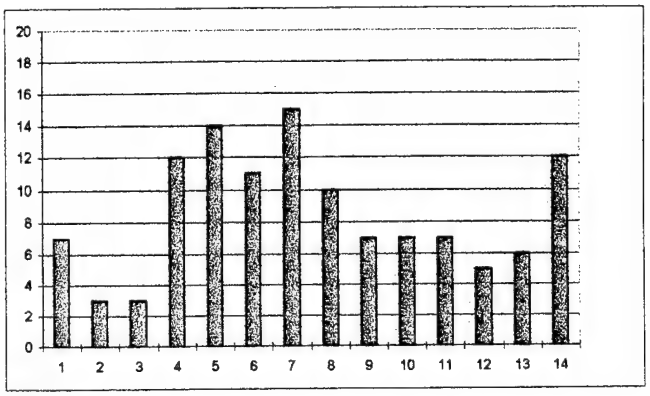


Figure 3.25. Histogram for Generation 2.

While there is still a great deal of diversity in the population, there is already a marked difference between this histogram and the one previous that came from a uniform distribution. Genes two and three seem to be converging to 0, but 1s were underrepresented in these genes in the initial population. Gene ten showed the most dramatic change, moving from having 13 1s initially to 7 in generation 2, a net change of 6. It will be seen if the trend continues in Generation 3's histogram:

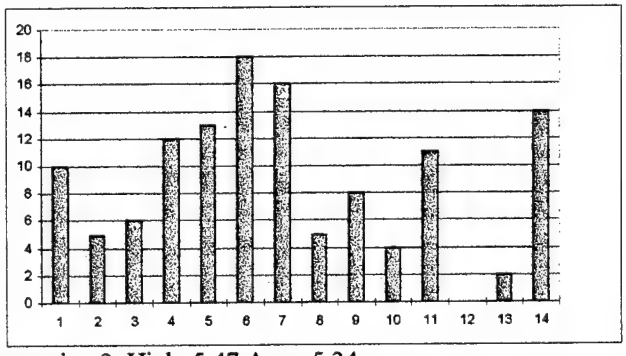
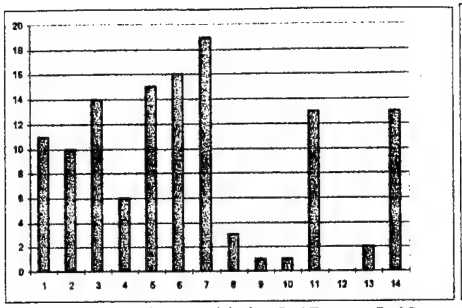
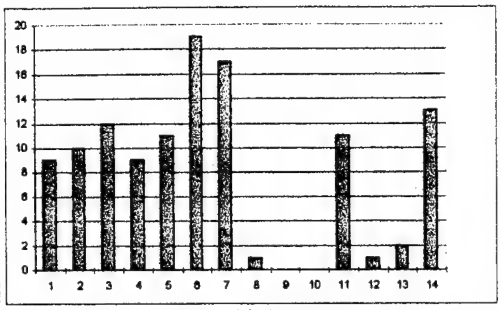


Figure 3.26. Histogram of Generation 3. High: 5.47 Avg.: 5.34

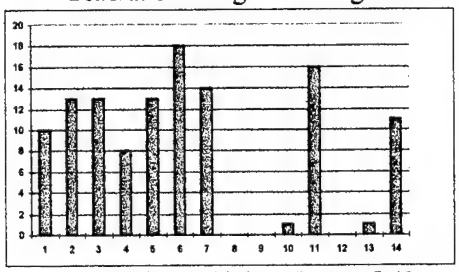
Notice how the fate of some genes seem to have reversed. Genes 2 and 3 are growing again, but Gene 12's diversity has been completely wiped out—only 0s remain in this gene in the population! Gene 13 as well moves toward 0, having only two 1s in the population. Gene 10 continues its move toward 0, dropping to 4. On the other hand, gene six has almost converged to all 1s—only two 0s remain in the population. No new top score was found, but the parent's average score for the next generation moved up to 5.34. Following is the histograms and important scores for the rest of the run.



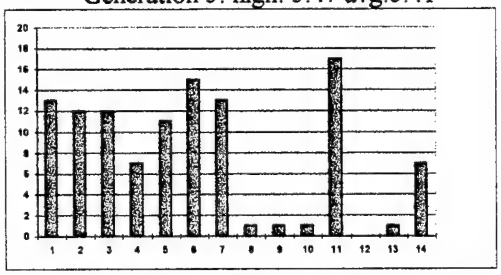
Generation 4: high: 5.47 avg:5.40



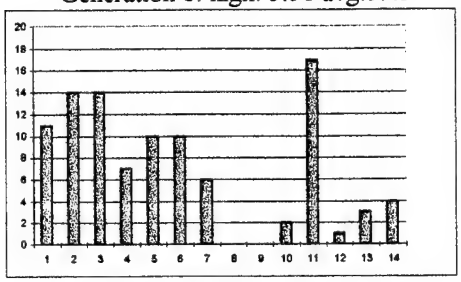
Generation 5: high: 5.47 avg:5.41



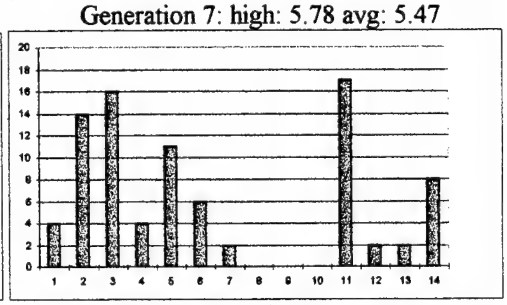
Generation 6: high: 5.51 avg:5.43



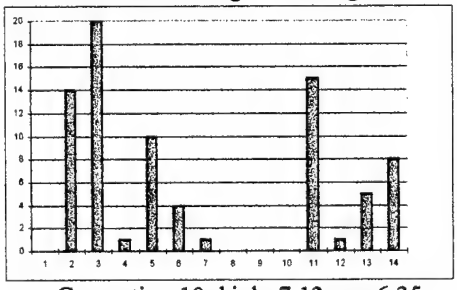
Generation 7: high: 5.78 avg: 5.47



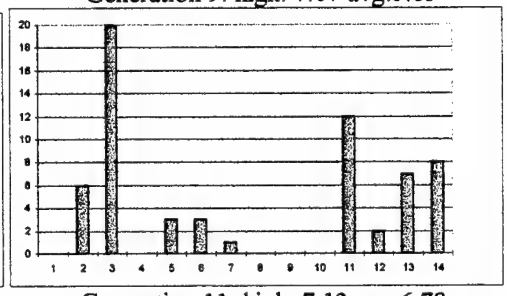
Generation 8: high: 6.44 avg:5.63



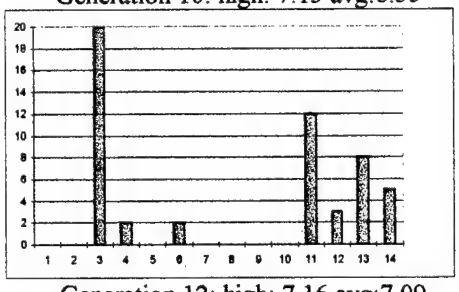
Generation 9: high: 7.07 avg:5.83



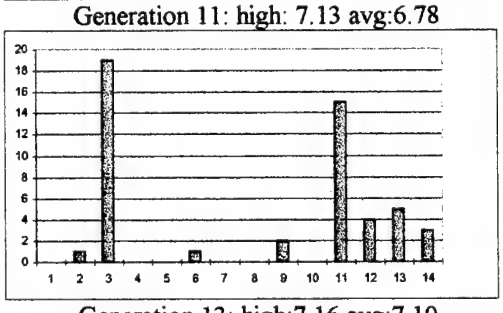
Generation 10: high: 7.13 avg:6.35



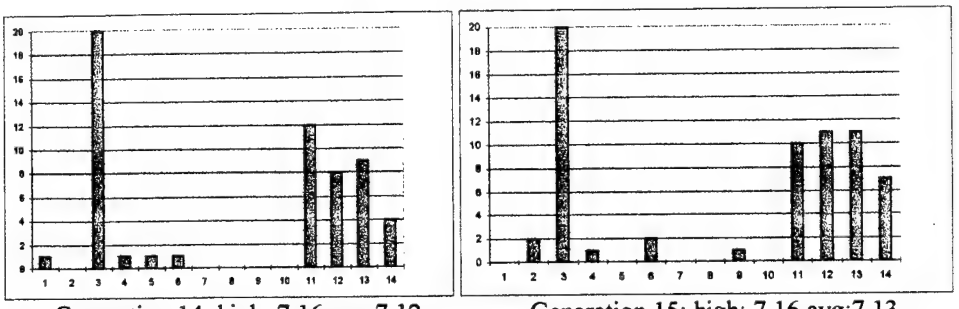
Generation 11: high: 7.13 avg:6.78



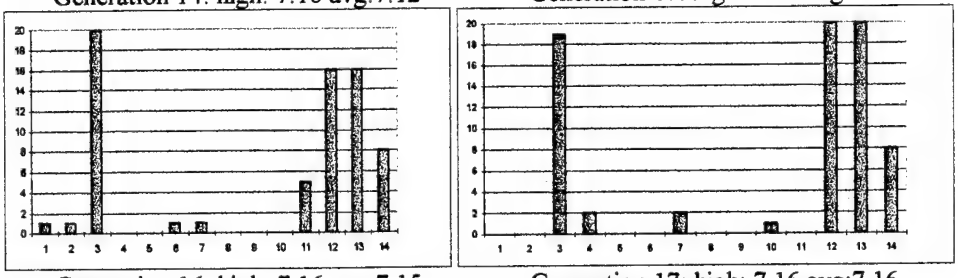
Generation 12: high: 7.16 avg:7.09



Generation 13: high:7.16 avg:7.10



Generation 14: high: 7.16 avg:7.12 Generation 15: high: 7.16 avg:7.13



Generation 16: high: 7.16 avg:7.15 Generation 17: high: 7.16 avg:7.16

Figure 3.27. Histograms from remainder of GA run

Here is the last generation's genepool:

Generation # 17

#1 00100000000110 sc: 7.160000
#2 00100000000110 sc: 7.160000
#3 00100000000110 sc: 7.160000
#4 00100000000110 sc: 7.160000
#5 00100000000110 sc: 7.160000
#6 00100000000111 sc: 7.130000
#7 00100000000111 sc: 7.130000
#8 00100000000111 sc: 7.130000
#9 00100000000111 sc: 7.130000
#10 00100000000111 sc: 7.130000
#11 00100000000110 sc: 7.160000 from 7 & 3, xo: 8
#12 00000010000111 sc: 3.850000 from 2 & 10, xo: 6
#13 00100000000110 sc: 7.160000 from 5 & 2, xo: 11
#14 00110000000111 sc: 5.320000 from 3 & 6, xo: 4
#15 00100000000110 sc: 7.160000 from 9 & 5, xo: 11
#16 00100000000110 sc: 7.160000 from 5 & 9, xo: 3
#17 00110000000111 sc: 5.320000 from 1 & 9, xo: 1
#18 00100000000110 sc: 7.160000 from 3 & 1, xo: 4
#19 00100010000110 sc: 6.570000 from 6 & 5, xo: 5
#20 00100000010110 sc: 4.230000 from 4 & 3, xo: 1

mutations: [12,7] [12,3] [14,4] [20,10] [16,14] [17,4] [19,7]
avg score: 7.16000

Notice that the diversity eventually drops to near zero for the whole population, and is zero in the parent population. The GA has been programmed to end once this convergence is reached. Note, however, that the GA found the best answer before convergence—in Generation 12. It required 146 NEC2 simulations to find the best answer, and 161 total NEC2 simulations for the entire run to go to convergence. Converting the top binary genotype, the decimal answer the GA found was: length 0.504λ, separation: 0.141λ.

As stated above, though, this is not the most typical run for this parameter set. Two other runs with the same parameters yielded top scores of 5.95 and 6.78. The first of these runs had a genotype of 01011110000110, which means the separation was the same as the "right" answer, but the length was 1.47λ, meaning that it was trapped into a local maximum.

The second run was also trapped into a local maximum, as shown by its genotype: 00100010000001. Its separation was small (0.0654λ), though the length was close to correct (0.535λ). The fact that the GA was stuck two out of three times into local maxima leads one to believe that the GA parameters are probably not optimized.

It is worthwhile to explore why a GA works now that one has been shown in detail, before moving on to the real-valued GA.

3.4.2 Why a GA works

This topic is covered in detail and excellently by Goldberg [22], so only a summary of his explanation will be stated here. But the analysis that has been presented thus far may imply that it is the individual gene values alone that are significant in a binary GA run. This is not the case. The theory behind the GA is as follows: in a binary GA, there are generally certain patterns within the genotype that are important for design performance. Consider the following very simple example:

Chrom.	Score
0111	16
1110	2
1010	5
1101	30

Table 3.2. Example scores for a simple problem.

In this example, the pattern ###1 seems to give a higher score, where "#" is a wildcard—it can match either a 1 or a 0. This pattern is an example of what has been termed a "schema." Other examples of schemata (the plural of schema) are 1##0, 110#, and even 1100. Note that a schema can have any assortment of the three characters {1,0,#}. In this way, there are 3^N possible schemata for a chromosome N bits long. Each chromosome is a member of 2^N schemata, as each gene can take on its specified value or the "#." Thus, each population with M chromosomes contains between 2^N and $M \cdot 2^N$ schemata at once, depending on the overlap between chromosomes.

In most well-constructed problems, it is the short, simple schemata with a small number of specified genes that have the most impact on a chromosome's performance. The presence of certain schemata in a chromosome can often ensure an above-average individual regardless of the other, less-important schemata it contains.

The operations of mating selections and crossover essentially exponentially promotes successful schemata, and exponentially demotes unsuccessful schemata, and this property is called the Fundamental Theorem of Genetic Algorithms by [22]. Because there are so many schemata present in the population, there are roughly M^3 schemata are usefully processed in each generation (for M or less cost function evaluations). In this way, many possible partial solutions are being processed at once. This parallel processing is called implicit parallelism.

One of the important consequences of this theorem is that in order for schemata to be most effectively processed, they need to be as small as possible. If they are spread out over the length of the chromosome, they are much more likely to be split as a result of crossover. Genes right next to one another have little chance of being split apart by a crossover point chosen to be between them. Genes at opposite ends of the chromosome will have to be split as a result of crossover, and processing is much more difficult. Thus, it is important to encode a chromosome to have the most highly related bits next to one another. This is why one will want to keep genes together that jointly describe a single number, like the first seven bits in this two-wire antenna problem that describe reflector length.

Now that the inner workings of the binary GA has been explored, the effect of changing the binary chromosome to a real one will be discussed.

3.4.3 The real-valued GA

The real-valued chromosome for this problem is only two genes long. It is convenient to have each gene only span the range from 0.0 to 1.0, to allow for consistent mating and child-creation, converting the real gene to its design value in a separate function. In this case, the conversion from L1 of the chromosome [L1 S1] to Length is:

Length = (L1)·4.

This allows Length to go from 0λ to 4λ. Similarly,

Separation = 0.05 + (S1)·1.95,

which allows Distance to span 0.05λ to 2λ.

Following is a GA run with the real chromosome, using Adewuya's mating and child-generation method. Mutation was Gaussian with a standard deviation of 0.1. There were 20 members in the population, 50% overlap from generation to generation, and 2.5% of the genes were mutated in each generation. (These are the same parameters as in the binary example.) Following is a run that converged to the right answer. The real GA was able to achieve a slightly higher score due to its ability to go to arbitrary accuracy.

Generation # 1
#1 0.8524 0.9442 sc: 3.230000
#2 0.2101 0.1615 sc: 3.560000
#3 0.4680 0.5392 sc: 4.050000
#4 0.2296 0.7861 sc: 3.680000
#5 0.6149 0.5960 sc: 4.880000
#6 0.2208 0.4255 sc: 3.650000
#7 0.8957 0.5301 sc: 3.900000
#8 0.9008 0.0562 sc: 5.460000
#9 0.4409 0.0798 sc: 5.270000
#10 0.0626 0.5251 sc: 3.860000
#11 0.5693 0.3800 sc: 4.330000
#12 0.1430 0.0845 sc:5.680000
#13 0.1716 0.7558 sc: 3.370000
#14 0.8760 0.3488 sc: 4.600000
#15 0.3802 0.3601 sc: 4.940000
#16 0.7372 0.7502 sc: 3.370000
#17 0.6845 0.4069 sc: 3.650000
#18 0.2077 0.8795 sc: 4.240000
#19 0.6991 0.2025 sc: 2.720000
#20 0.2372 0.6458 sc: 4.160000
Top 50% avg score: 4.76100

Following is a scatter plot showing the positions of these first 20 individuals in the search space. Notice that this is fundamentally different from the binary histogram, which showed the amount of each gene in the population, and that the scattering is random throughout the space.

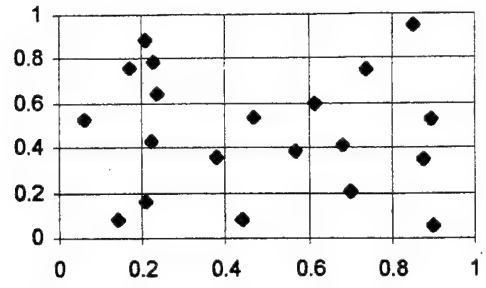


Figure 3.28. Scatter plot for Generation 1. The "Length" gene is plotted on the horizontal axis, "Separation" gene is on the vertical axis. Best chromosome has score 5.68, top 50% avg: 4.76

In the mating process to create Generation #2, 30% of the children's genes were created through quadratic crossover, and 70% through heuristic.

Generation # 2
#1 0.1430 0.0845 sc: -5.680000
#2 0.9008 0.0562 sc: -5.460000
#3 0.4409 0.0798 sc: -5.270000
#4 0.3802 0.3601 sc: -4.940000
#5 0.6149 0.5960 sc: -4.880000
#6 0.8760 0.3488 sc: -4.600000
#7 0.5693 0.3800 sc: -4.330000
#8 0.2077 0.8795 sc: -4.240000
#9 0.2372 0.6458 sc: -4.160000
#10 0.4680 0.5392 sc: -4.050000
#11 0.6468 0.1724 sc: -3.070000 from 2 & 7 & 1
#12 0.1728 0.9958 sc: -3.450000 from 7 & 8 & 6
#13 0.2038 0.0053 sc: -5.030000 from 3 & 9 & 4
#14 0.6923 0.4087 sc: -3.620000 from 4 & 3 & 2
#15 0.4079 0.2181 sc: -2.570000 from 4 & 3 & 1
#16 0.6630 0.4286 sc: -3.240000 from 7 & 5 & 3
#17 0.9948 0.3697 sc: -4.420000 from 3 & 6 & 5
#18 0.6037 0.5568 sc: -4.490000 from 3 & 6 & 4
#19 0.4790 0.8592 sc: -4.490000 from 3 & 7 & 10
#20 0.0207 0.3024 sc: -3.850000 from 2 & 9 & 5
Mutations: [16,1]
Top 50% avg score: -4.92600

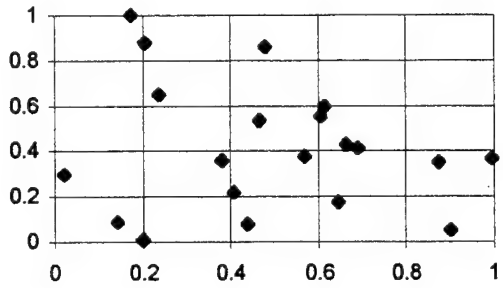
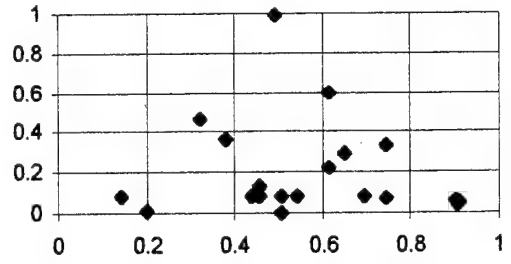
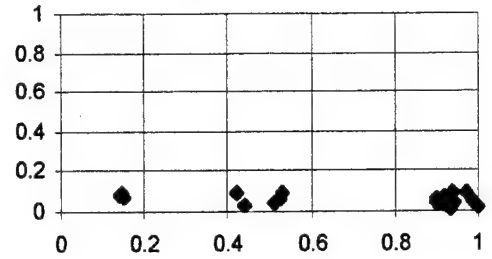


Figure 3.29. Scatter plot for Generation 2. The "Length" gene is plotted on the horizontal axis, "Separation" gene is on the vertical axis. Best: 5.68, top 50% avg: 4.93

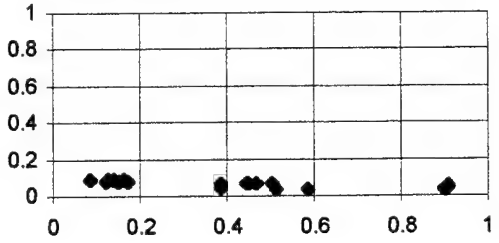
Following are the plots for every third generation up to generation 23, after which the individuals are tightly clustered around the same point shown in Generation 23. Notice that the separation variable converged onto the correct value by only Generation 6. This is not surprising since the optimal separation is roughly the same even if a chromosome specifies a different length. Also, the hill is larger for the separation variable than for the length variable.



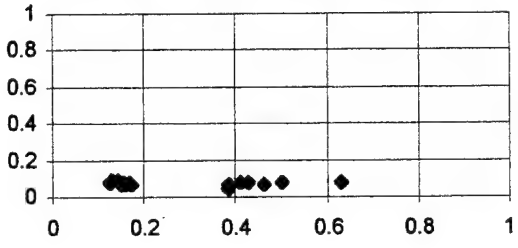
Gen 5—best: 5.68, top 50% avg: 5.36



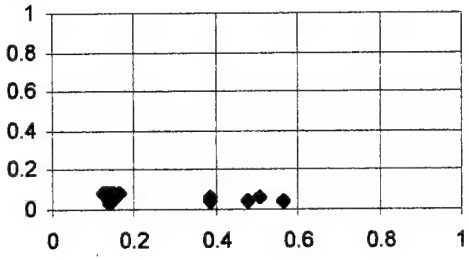
Gen 8—best: 5.68, top 50% avg: 5.51



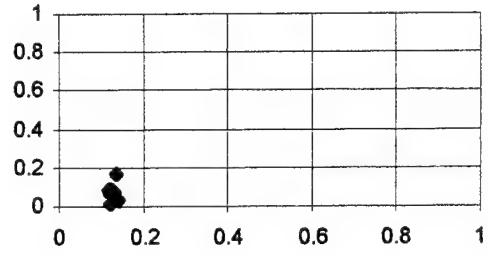
Gen 11—best: 7.00, top 50% avg: 5.85



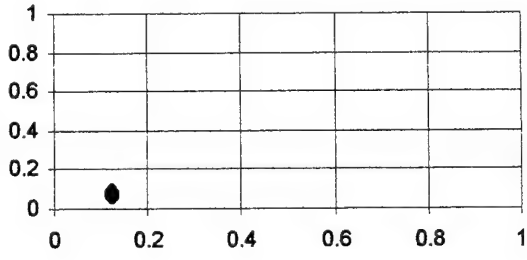
Gen 14—best: 7.00, top 50% avg: 5.98



Gen 17—best: 7.00, top 50% avg: 6.19



Gen 20—best: 7.04, top 50% avg: 7.03



Gen 23—best: 7.19, top 50% avg: 7.13.

Figure 3.30. Scatter plots for remainder of GA run. The "Length" gene is plotted on the horizontal axis, "Separation" gene is on the vertical axis.

Plots for Gen. 26 and 29 are essentially the same as for Gen. 23. Gen. 26 had a best score of 7.19, and a top 50% average of 7.14. Gen. 29, the last generation, had a best score of 7.19, and a top 50% average of 7.19.

In the final generation, quadratic crossover did not occur, only heuristic, because only one individual was mated. This single-chromosome selection occurred because of fitness scaling. 7.14 was the score of every other chromosome, so the fitness of every other chromosome but #1 was 0.0, while the fitness of #1 was 0.05. This gave the entire wheel to chromosome #1, and shows a slight weakness in this scaling scheme. When the GA converges almost completely, save one better chromosome, that chromosome will dominate the mating of the next generation. However, this only occurs when the GA is almost completely converged anyway, and so is not worrisome. In addition, there was no successful quadratic crossover in the last 4 generations, showing the GA was fairly converged even before the final generation.

Generation # 29
#1 0.1236 0.0623 sc: 7.190000
#2 0.1223 0.0792 sc: 7.140000
#3 0.1220 0.0799 sc: 7.140000
#4 0.1223 0.0786 sc: 7.140000
#5 0.1222 0.0795 sc: 7.140000
#6 0.1220 0.0799 sc: 7.140000
#7 0.1221 0.0797 sc: 7.140000
#8 0.1223 0.0792 sc: 7.140000
#9 0.1222 0.0795 sc: 7.140000
#10 0.1222 0.0795 sc: 7.140000
#11 0.1236 0.0058 sc: -9.280000 from 1 & 1 & 1
#12 0.1236 0.0623 sc: 7.190000 from 1 & 1 & 1
#13 0.1236 0.0623 sc: 7.190000 from 1 & 1 & 1
#14 0.1236 0.0623 sc: 7.190000 from 1 & 1 & 1
#15 0.1236 0.0623 sc: 7.190000 from 1 & 1 & 1
#16 0.1236 0.0623 sc: 7.190000 from 1 & 1 & 1
#17 0.1236 0.0623 sc: 7.190000 from 1 & 1 & 1
#18 0.1236 0.0623 sc: 7.190000 from 1 & 1 & 1
#19 0.1236 0.0623 sc: 7.190000 from 1 & 1 & 1
#20 0.1236 0.0623 sc: 7.190000 from 1 & 1 & 1
mutations: [11,2]
avg score: -7.19000

The "correct" answer took 209 NEC2 simulations to achieve, and convergence occurred in 284 simulations. Notice this is almost twice as many computations as the binary GA required. However, as will be seen in Chapter 5, some problems are only effectively solved with a real GA.

Also, other real-chromosome GA runs with these parameters did not produce these results. These other runs achieved 5.96, 5.96, 5.37, 5.50 and 5.53 dB of gain—much worse than the one described here. In addition, they were all trapped into local maxima. The first two had approximately 1.6λ length and 0.15λ separation. The next converged to 3.66λ length and 0.08λ separation. The last two converged to 1.92λ length, 0.14λ separation and 2.33λ length, 0.13λ separation respectively. Looking at the response surface, it is easy to see that each time the GA was trapped in the local maxima at those locations. Obviously, then, the GA had very inappropriate parameters.

3.5 Optimizing the example GA

As was mentioned above, each example GA shown above was not optimal in its parameters. The parameters were chosen so that the analysis would be simple and a listing of the whole population would not be too lengthy. However, a natural question to ask is what parameter set is optimal for this GA. To answer that question, an experiment was

conducted to optimize this GA in both its binary and its real-valued forms. Three variables were studied: population size, percent of population overlap, and mutation rate.

Population size had 6 values: 10, 20, 40, 80, 160, and 320. Percent overlap had three values: 0.3, 0.5, and 0.7. Mutation rate had 5 values: 5%, 2.5%, 1.25%, 0.625%, and 0.3125%.

Because this was a full-factorial experiment, statistical methods could be applied to find the optimal parameters. However, it is of use just to do some basic analysis of the performance of the GA. More involved analyses occur in later chapters, where precise optimization is more important.

First, the binary case was explored. The 90-run experiment (6 population sizes ·3 percentage overlap levels ·5 mutation rates) was repeated 3 times to give a better representation of the behavior of the GA.

The most important variable by far was population size. The effect of the other two variables paled in comparison. Averaged over all mutation rates and percentages, the following table shows the effect of population size:

Population size	Average score	Standard Deviation	% that find the "right" answer
10	5.68	0.48	6.7
20	6.14	0.40	13.3
40	6.77	0.24	55.6
80	6.97	0.25	75.6
160	7.12	0.081	86.7
320	7.16	0.000	100

Table 3.3. Binary GA performance vs. Population size

So it would appear that a large population is best. The problem is that the number of runs to convergence increases dramatically with population size, as shown below.

Population size	Average NEC2 sims to top score	St Dev of NEC2 sims to top score	Average NEC2 sims to converge	St Dev NEC2 sims to converge
10	20.44	8.25	26.1	10.8
20	55.4	19.9	69.5	27.2
40	121.47	27.4	149.1	47.1
80	235.5	59.0	308.6	104.6
160	506.62	178	645.5	250.1
320	748.6	117.0	1096.9	275.4

Table 3.4. NEC2 simulations vs. Population size for a binary GA

As is shown, then, one pays a price for the security of knowing one has the right answer. What is the best situation for a given problem depends on the time allowed and the importance of being certain a given answer is the absolute best available.

A similar situation arose in the real-valued GA.

Population size	Average score	Standard deviation	% that find the "right" answer
10	5.54	0.24	3.8
20	5.96	0.25	11.1
40	6.37	0.21	30.5
80	6.77	0.17	61.0
160	7.05	0.16	86.5
320	7.16	0.061	97.1

Table 3.5. Real GA performance vs. Population size

Population size	Average NEC2 sims to top score	St Dev of NEC2 sims to top score	Average NEC2 sims to converge	St Dev NEC2 sims to converge
10	133.7	98.2	253.8	101.4
20	185.6	85.1	264.6	91.9
40	349.6	83.6	432.6	83.2
80	609.5	120	756.6	120.5
160	1043	237	1311	290.8
320	1797	231	2420	353.1

Table 3.6. NEC2 simulations vs. Population size for a real GA

As can be easily seen, one not only pays for the security of the best answer with more NEC2 runs, but the real GA requires a much larger number of runs for all population sizes. This has been found to be the case for all the antennas optimized in this thesis. However, some problems have much more difficult and sensitive optima, and the real-numbered GA seems to be able to find them in less time, given all the repeated runs necessary for the binary GA to find similar optima. The real GA also found better answers than the binary GA because of its arbitrary accuracy. The real GA does not necessarily supersede the binary GA for all cases, however, as the improvement in score and reliability may not always justify the higher cost of the simulation time. More differences will be discussed in later chapters as different antennas are optimized with both real and binary GAs.

As the effect of the other two variables—mutation rate and overlap—were more subtle, their effect will be explored in later chapters where more complete experiments are described. However, their effect is almost negligible compared to the overriding effect of population size. De Jong performed a similar investigation in his 1975 thesis for the simple GA [30].

Note that, though all the parameters were varied greatly in both the real and binary GAs, they did not require that one use the best possible parameters in order to obtain the correct answer at least occasionally. This shows clearly the robustness of the GA to its parameters.

3.6 Conclusion

This chapter contains a great deal of GA information, primarily practical in nature, and an example of a normal binary and real GA run. In succeeding chapters, GA optimizations of various antenna designs will be explored, and the process of finding the optimal parameters for the GA optimizing those designs.

Chapter 4: The Loaded Monopole

4.1 Introduction

A monopole loaded with a modified folded dipole, as is seen in Figure 1, has been previously investigated. It was shown that when the inserted folded element is approximately 0.1λ above the ground plane and the height of the monopole is about 0.35λ, then the E_θ-pattern in the plane of the folded element (i.e., the cut at $\phi = 0°$) approaches hemispherical coverage [1]. In this chapter, a GA is utilized to optimize the above configuration for uniform power throughout the upper hemisphere (i.e., for all ϕ and where $-90° \leq \theta \leq 90°$). This was the first wire antenna to be optimized for this research, and, to the best of my knowledge, was the first antenna composed only of wire to be optimized with a GA.

The GA is applied in the procedure described in Chapter 3. Each of the wires that make up the antenna is designated to have a range of possible lengths. The GA randomly selects the initial population. As with the example antenna of Chapter 3, the GA program converts each configuration from its internal representation (explained in the next section) to a file that can be read by NEC2. The GA then calls NEC2, which computes the sample's radiation pattern and places the results in an output file. The GA program reads the output file, and the radiation pattern is compared with the desired pattern and scored. As will be discussed, the optimization for this antenna has a single goal: a sum-of-squares match to a uniform, broad-beam pattern.

This antenna was optimized using a binary GA. An experiment was performed to optimize the binary GA parameters. Since this antenna has only six unknowns, an exhaustive search was conducted of the search space to explore its characteristics. After these were performed, the antenna was optimized with a real chromosome to compare the results. One of the earliest high-quality outputs of the binary GA optimization was fabricated and tested to verify that the simulation matched reality. Such verification was useful to ensure that the GA results were valid and not exploiting some weakness in the simulator.

4.2 The search space

In order to implement a GA, it is necessary to select a set of possible lengths for each wire of the loaded monopole. It is important that each range of lengths be large enough that the optimal length is likely to be included, yet not so large that the search space becomes unmanageable. Also, the binary-string GA requires the range of lengths for each wire to be broken into an integral number of different possible lengths from which the GA can choose.

The figure below shows the search space for the loaded monopole antenna. Note that it only has six unknowns, two X-coordinates and four Z-coordinates. The ranges for each variable are shown in the figure. These ranges were chosen using the prior results and experience obtained in [1], though they were relaxed somewhat beyond what would have normally have been considered. In addition, separating the two X-coordinates to allow asymmetry had not been explored before, but turned out to be crucial to the performance of the antenna.

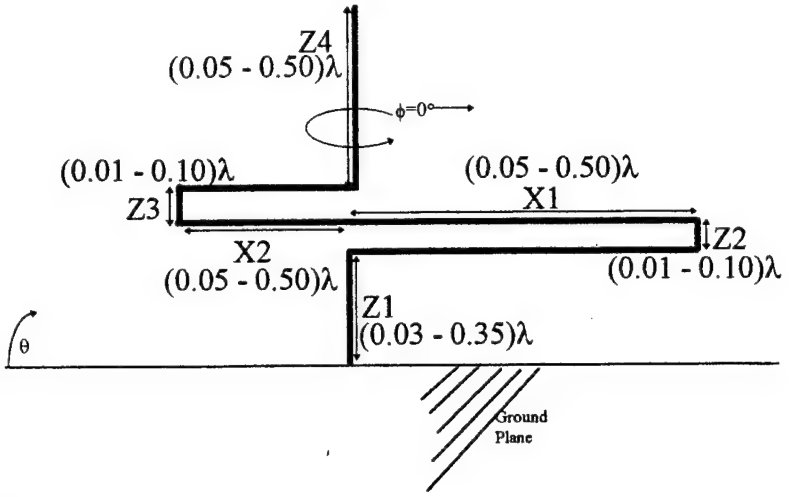


Figure 4.1. Monopole antenna loaded with a modified folded dipole. Numbers in parentheses indicate initial range of lengths.

Each wire is represented by a 5-bit string and thus has 32 possible lengths. Five bits per variable were chosen because it gave a resolution (i.e., a lower bound on the smallest change) of 0.014λ to 0.003λ (depending on the wire), which was on the order of the fabrication tolerance. This resolution gave very good results. If adequate results had not been obtained, resolution would have been increased or decreased. Increasing the resolution would have been valuable if the bandwidth was inherently narrow and it was needed to fine-tune the wire lengths, but would have increased the size of the search space and made the GA's job more difficult. Decreasing the resolution would have enabled the algorithm to search the space more quickly and/or more exhaustively (since there would be fewer solutions to choose from), but there would be a risk that good solutions would no longer exist in the coarser search space.

4.3 The objective function: optimization for wide beamwidth

The radiation pattern of each individual was computed using NEC2, compared with the desired pattern (uniform power over the hemisphere for this antenna) using a cost function containing a least mean square criterion, and ranked according to performance.

The specific cost function for this antenna is:

$$\text{Score} = \Sigma_{\text{over all } \theta,\phi}(\text{Gain}(\theta,\phi) - \text{Avg. Gain})^2$$

Unlike the example in Chapter 3, the objective of the optimization was to minimize this score. A perfect antenna would have a score of zero.

4.4 Initial GA results

The GA was repeated a number of times for the loaded monopole, and, while it never produced identical designs, they were usually similar in their dimensions. It is not surprising that the antennas it produced were never identical, since there are over one billion possible designs with the 30-bit binary chromosome, and the GA is a probabilistic strategy, including a random initial population, and random selection of crossover points for mating chromosomes.

For the initial GA run of this antenna, a population of 150 binary chromosomes was used, with an overlap of 50%. The exact mutation rate that varied from generation to generation, as the GA was allowed to flip a random number of bits totaling between 1 and 20 in the children in each generation. One ϕ angle of 0°, and 181 θ-angles (-90° to 90°, in

1° increments) to get the sum-of-squares variation over the hemisphere was used. As will be described, though, this antenna optimization was run with many different GA parameters, though always with the same cost function.

A fitness-weighted roulette wheel was used for mating selection, with a fitness transformation. The transformation was the same one used in all optimizations in this thesis, and was necessitated by the fact that this was a minimization instead of a maximization. The fitness transformation, in which the scores are all scaled relative to the score of the worst individual, is described in Chapter 3. Single-point crossover was used as well.

The score for the antenna resulting from this initial GA was 30.9 for $\phi = 0°$, 45°, and 90°, and 13.7 for just the $\phi = 0°$ cut. This is among the best that was found by all GA optimizations of this antenna. It had an excellent simulated pattern, and an unusual, asymmetric shape—not at all what was expected. It was built and tested, and thus its measurements and characteristics are discussed in the section in this chapter entitled "Antenna validation and testing." Suffice to say here that it was a very encouraging result.

4.5 Optimizing the GA

Though the initial results were quite promising, it was desired to know whether the GA was itself optimized. A full-factorial designed experiment was performed to optimize the GA for the loaded monopole. The variables included were: number of ϕ-angles (designated "# Ang" in the following table), population size (M), and percent overlap (%). Mutation rate was not included in this experiment, and was allowed to randomly vary between 1 and 20 bit-flip mutations per generation. The number of θ-angles used in each simulation was held constant at 181 (from -90° to plus 90°, including 0°), so changing the number of ϕ-angles would change the number of total angles in multiples of 91.

Following is the runsheet for this experiment. The runsheet is included here for illustration; future experiment runsheets are similar and will not be shown. As one can see, each variable had two levels. The number of ϕ-angles varied between 1 to 3 (0° only, versus 0°, 45° and 90°), population between 50 and 500, and overlap between 30% and 70%. The full experiment was replicated eleven times, so eleven data points are shown for each combination of parameters. Listed are the number of generations required to converge, the number of NEC2 simulations needed to achieve convergence, the best, raw scores found by the GA after convergence, and the performance of each optimized design based on three ϕ-angles (the more accurate measure of hemispherical coverage).

			Generations required to converge											avg	s
#Ang	M	%	1	2	3	4	5	6	7	8	9	10	11		
1	50	0.7	29	26	19	37	18	39	64	28	46	19	28	32.09	13.80
1	50	0.3	7	14	8	5	9	12	7	10	7	5	8	8.36	2.77
1	500	0.7	23	20	22	20	36	26	18	16	15	27	21	22.18	5.90
1	500	0.3	7	10	8	11	9	11	12	8	11	16	6	9.91	2.77
3	50	0.7	35	22	22	16	18	29	17	20	33	28	22	23.82	6.48
3	50	0.3	15	8	14	14	12	15	5	5	4	10	7	9.91	4.30
3	500	0.7	22	65	17	23	15	22	19	18	17	36	15	24.45	14.67
3	500	0.3	9	15	14	9	10	9	9	14	11	11	11	11.09	2.26

			Number of NEC calls											avg	s
#Ang	M	%	1	2	3	4	5	6	7	8	9	10	11		
1	50	0.7	498	450	338	626	322	658	1058	482	770	338	482	547.45	220.83
1	50	0.3	266	518	302	194	338	446	266	374	266	194	302	315.09	99.60
1	500	0.7	3822	3369	3671	3369	5785	4275	3067	3765	2614	4426	3520	3789.36	834.78
1	500	0.3	2606	3659	2957	4010	3308	4010	4361	2957	4010	5765	2255	3627.09	973.41
3	50	0.7	594	386	386	290	322	498	306	354	562	482	386	415.09	103.65
3	50	0.3	554	302	510	518	446	554	194	194	158	374	266	370.00	154.06
3	500	0.7	3671	10164	2916	3822	2614	3671	3218	3067	2916	5785	2614	4041.64	2215.50
3	500	0.3	3308	5414	5063	3308	3659	3308	3308	5063	4010	4010	4010	4041.91	791.96

			Best Scores after convergence, as optimized, using 91 or 273 angles											avg	s
#Ang	M	%	1	2	3	4	5	6	7	8	9	10	11		
1	50	0.7	6.99	12.7	31.9	0.739	42.2	49.35	11.9	9.18	98.2	17.4	95.8	48.40	88.13
1	50	0.3	50.7	16.7	135.7	50.1	49.35	9.18	98.2	17.4	95.8	85.5	50.4	59.91	39.73
1	500	0.7	5.12	10.2	5.11	1.68	4.33	2.99	25.6	5.1	10.83	10.46	3.79	7.75	6.70
1	500	0.3	8.9	0.884	7.79	6.81	1.09	6.31	1.99	11.2	4.8	1.2	13.73	5.88	4.36
3	50	0.7	52.4	124	62.7	283	340	64.5	133	139	21.5	66.4	520	164.23	154.11
3	50	0.3	115	147	116	529	64.5	437.6	741	130	439	50.1	98.7	260.72	233.89
3	500	0.7	10.6	64	18.9	24.2	53.7	32.2	32.9	14.2	37.9	36	41.2	33.25	16.19
3	500	0.3	23.7	29.5	48.3	30.6	29.3	23.3	23.4	27.5	33.1	21.8	26.3	28.80	7.38

			Hemispherical performance, using 273 angles in all cases											avg	s
			1	2	3	4	5	6	7	8	9	10	11		
1	50	0.7	33.4	122	1293	317	100	196	1944	1377	773	267.8	249.3	606.59	647.76
1	50	0.3	91.9	232	263	126	250	43.6	224	94.1	424	182	1652	325.69	452.32
1	500	0.7	49.8	78	273	494	48.5	479	60.9	29.2	1703	27.2	9.18	295.62	500.32
1	500	0.3	854.5	473	744	231	367	35.7	432	24.6	99.7	766	30.9	368.95	313.74
3	50	0.7	52.4	124	62.7	283	340	64.5	133	139	21.5	66.4	520	164.23	154.11
3	50	0.3	115	147	116	529	64.5	437.6	741	130	439	50.1	98.7	260.72	233.89
3	500	0.7	10.6	64	18.9	24.2	53.7	32.2	32.9	14.2	37.9	36	41.2	33.25	16.19
3	500	0.3	23.7	29.5	48.3	30.6	29.3	23.3	23.4	27.5	33.1	21.8	26.3	28.80	7.38

Legend:
#Ang = # of Phi angles included in the optimization
M = Population size
% = Percent of population saved betw. generations
avg = average
s = est. std. Dev.

Table 4.1. Run sheet for GA parameter experiment

Notice that, as in the example GA in Chapter 3, the population size varied dramatically during the experiment. Even so, the GA was occasionally able to find a good answer occasionally with a small population. Of course, the GA was much more reliable with a large population. Notice that, as in Chapter 3, a price is paid to ensure the GA converges to a good answer: the number of runs required for large population sizes is an order of magnitude larger than with small ones. One would probably want to consider running small populations repetitively instead of putting all hope into a single GA run given these results. To find just one answer below 50, one would have to run the GA with a small population approximately 15 times, with an expected cost of 6200 NEC2 simulations. One would have a 91%

chance of getting a similar answer out of only one large-population GA run, with about 4000 NEC2 simulations. But if one did not get the right answer, one would have to run the GA again, with another 4000 NEC2 simulations!

Overlap did not seem to make a large difference. Sometimes the average optimal score with an overlap of 0.3 would be smaller, sometimes larger than the 0.7 overlap. Thus, it turned out to be a weak predictor of GA performance. This variable is explored more deeply with the crooked-wire antenna, to be discussed in Chapter 6.

While it may be obvious why population size and overlap were varied, it may not be as clear why the number of ϕ angles were changed. Several preliminary runs were conducted using only one ϕ angle, including the one that was eventually built and tested, thinking that if one ϕ-cut were optimal, that might translate into all ϕ-cuts being optimal. If this had been true, it would make the GA more sensitive to small variations in individual datapoints. Having a large number of angles to sum over could mean that a small amount of variation over a large pattern could have the same score as a very even pattern with a single unacceptable, narrow null.

It turned out, however, that the performance of the GA changed dramatically with the number of ϕ angles, and that a larger number of angles was important to have. Though the raw GA scores were not very different, the individuals produced by only one ϕ-cut had much poorer patterns over the hemisphere. This was important to learn: angles were important to sample frequently in an antenna pattern. Fortunately, not all quantities, especially frequency dependence, are so sensitive, as will be shown in the next chapter.

4.6 Exhaustive search results

The loaded monopole search space was further investigated to determine the suitability of a GA as opposed to classical methods (e.g., gradient methods) of optimization. This exhaustive study, involving over 65,000 NEC2 runs, spanned the whole search space (though at lower resolution than the GA optimization).

The four larger variables X1, X2, Z1, and Z4 all received 8 levels. The other two, Z2 and Z3, since they have much smaller ranges, had only 4 levels each. All told, there were 65,536 different combinations which were simulated.

The results of this search are difficult to visualize, because, unlike the example of Chapter 3, there are six unknowns, and only a few levels for each, meaning that surfaces showing the relationship between only two variables are incomplete and low-resolution. However, it is of interest to see the relationship of each variable alone on the antenna score. Hence, the following series of histograms only show just one variable at a time, with all scores in a row above the valid values of the variable. Z1 will be examined first.

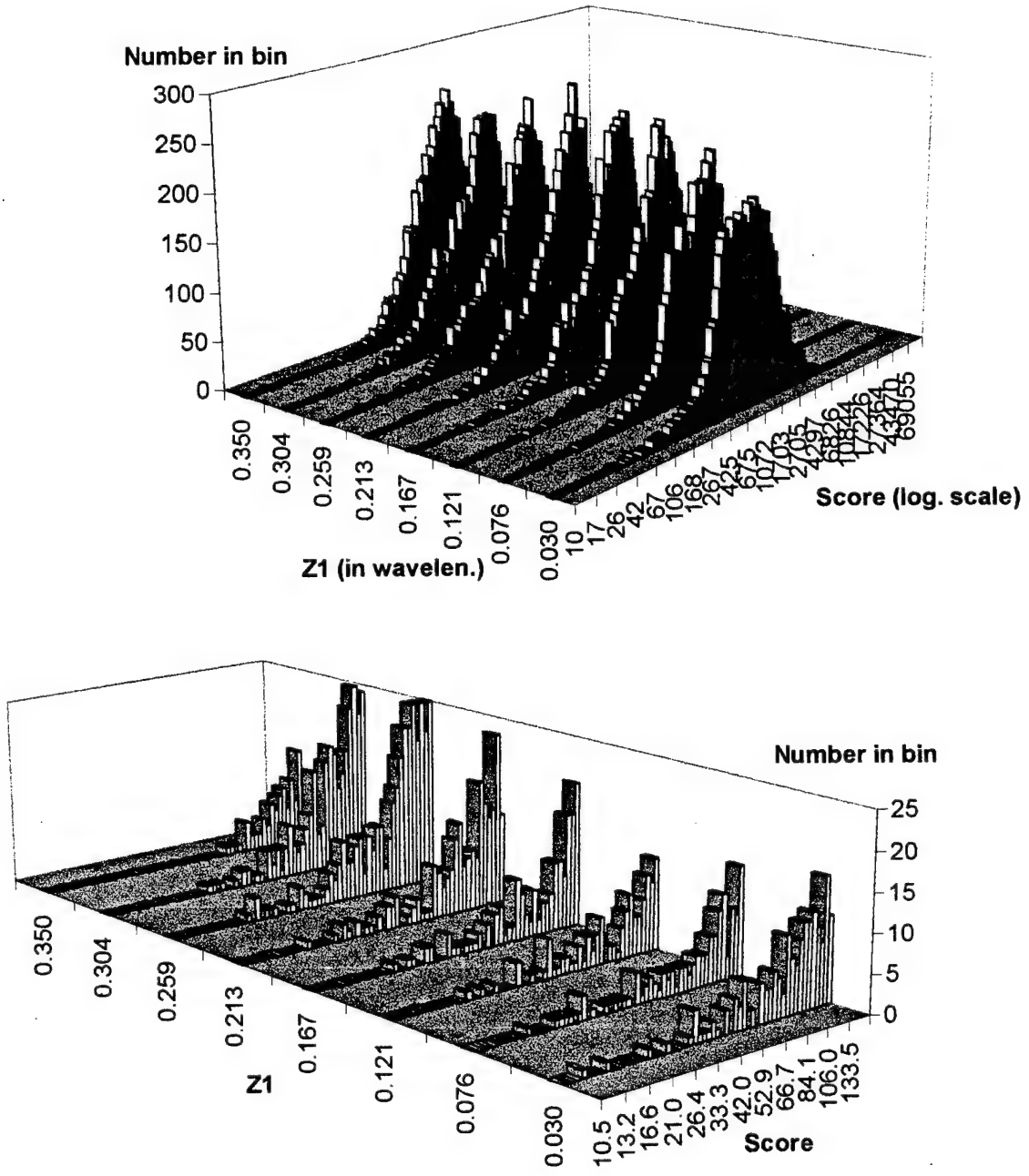


Figure 4.2. Z1 Histogram and close-up histogram of the low-score end.

Note that the distributions of all the histograms for all values of Z1 cover almost the whole range of scores, from poor to reasonably good. Thus, Z1 is not a particularly good predictor of performance by itself: given any value of Z1, a good individual could result if the other parameters were properly chosen. When the score is plotted on a logarithmic scale, they resemble normal distributions, and the means are very similar. The standard deviations differ slightly, though, and seem to increase as Z1 decreases. In the close-up view, one can see that having a lower Z1 value makes it possible for a better result, however, good results are obtainable even with Z1 values as high as 0.304.

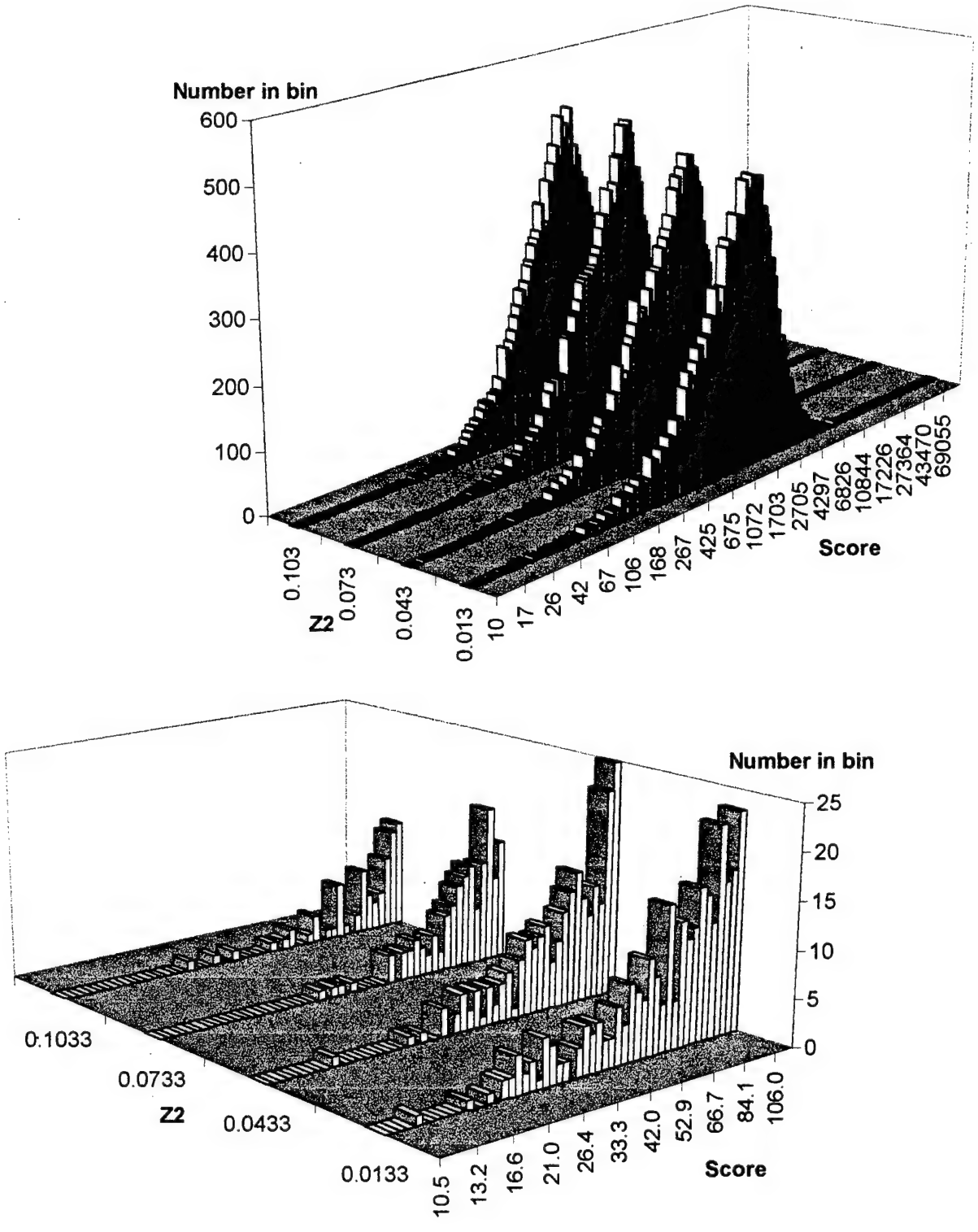


Figure 4.3. Z2 Histogram and Close-up of low score

Z2 has only four levels, and is apparently not a sensitive factor by itself. All values of Z2 have good antennas in their distributions. Note that, again, the distributions are normal-like and have similar means and standard deviations.

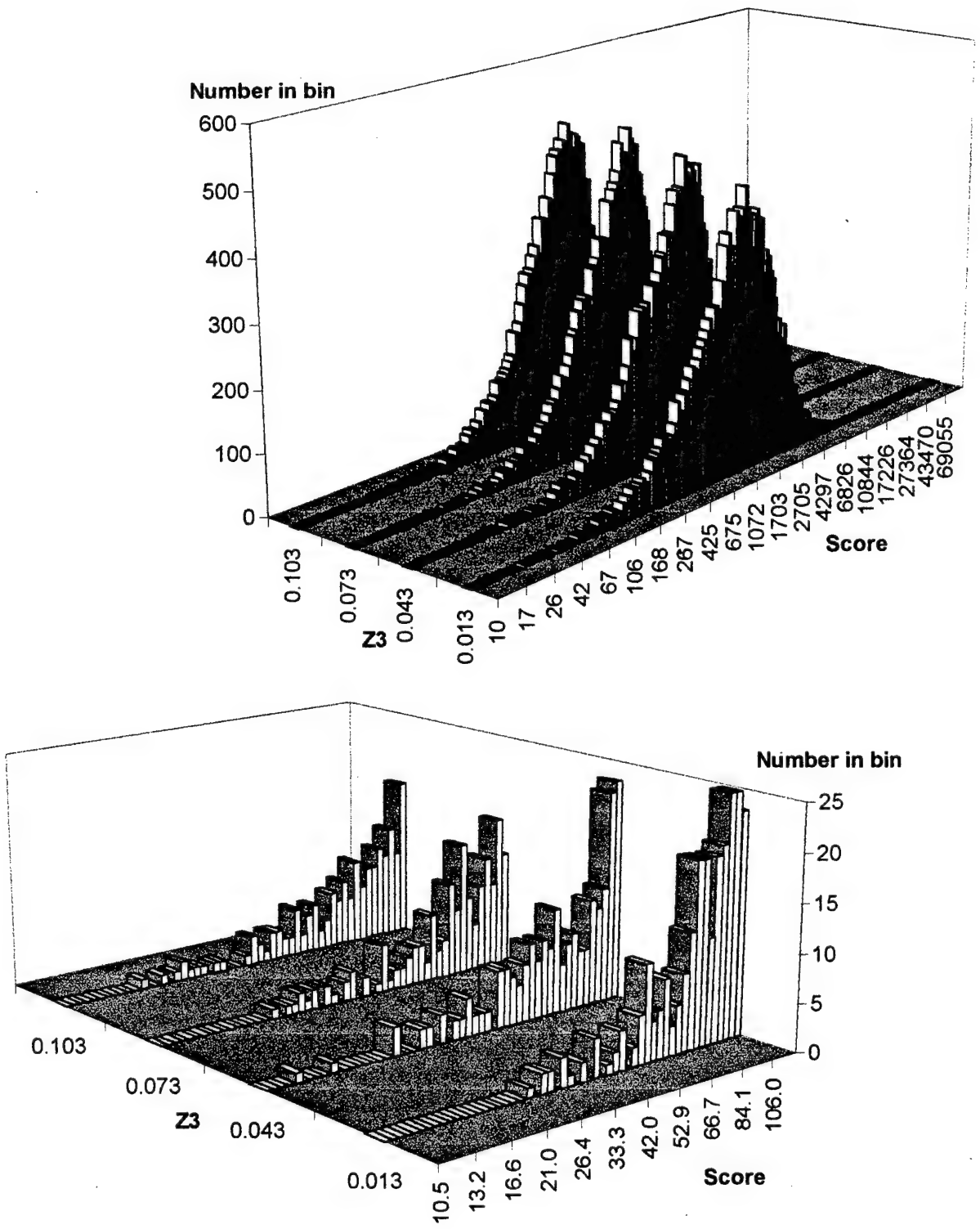


Figure 4.4. Z3 Histogram and Close-up of low-score end.

Again, Z3 shows a similar pattern to Z2, however, the best values seem obtainable only with the higher three values of Z3. Even so, a value of 0.013 for Z3 can produce good results.

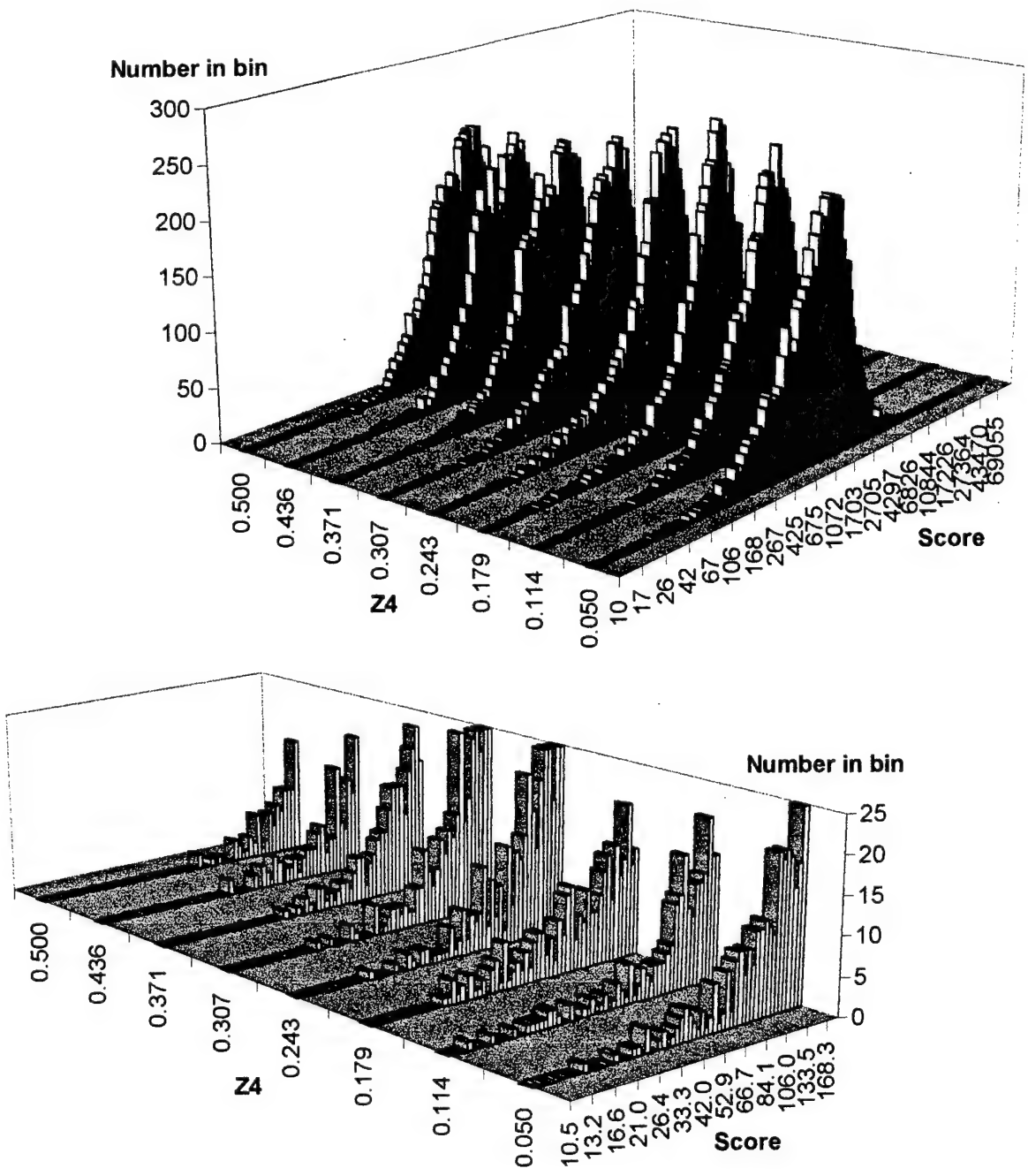


Figure 4.5. Z4 Histogram and close-up of low-score end.

Z4's histograms seem similar to Z1's. In Z4's case, the best scores seem to be at 0.114, and higher values of Z4 limit the score to some extent.

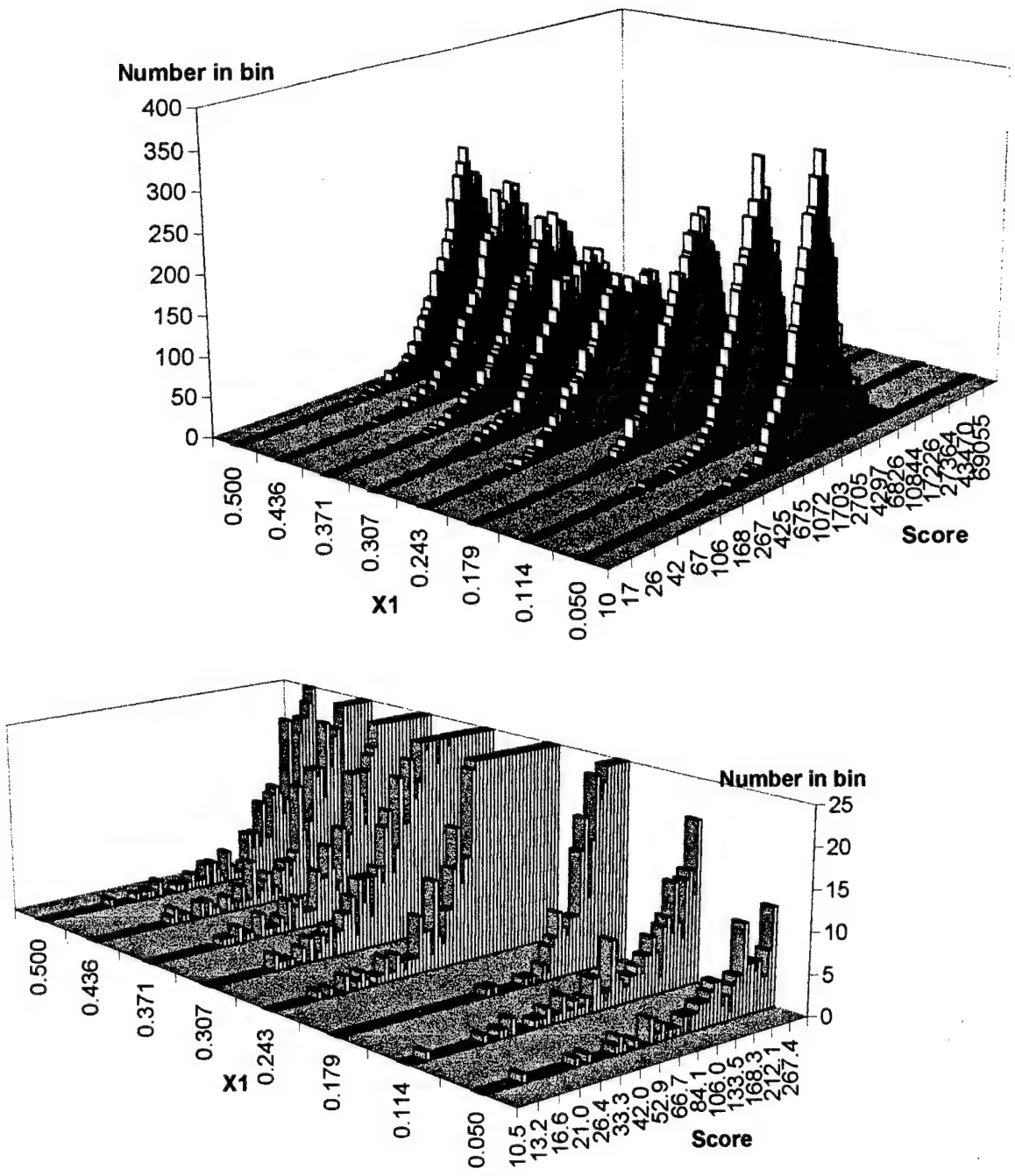


Figure 4.6. X1 Histograms and Close-up of low-score end.

X1 has distributions with about the same mean, but they have different standard deviations. Though there is a larger standard deviation when X1 is 0.243, the best score is achieved with X1=0.114. Note that there is a gap in good scores at X1=0.179. This seems to imply that the good scores obtained at X1=0.243 are not on the same hill as those obtained with X1 $\leq$ 0.114, because if they were, there should be scores for X1=0.179 that are between those at 0.243 and 0.114.

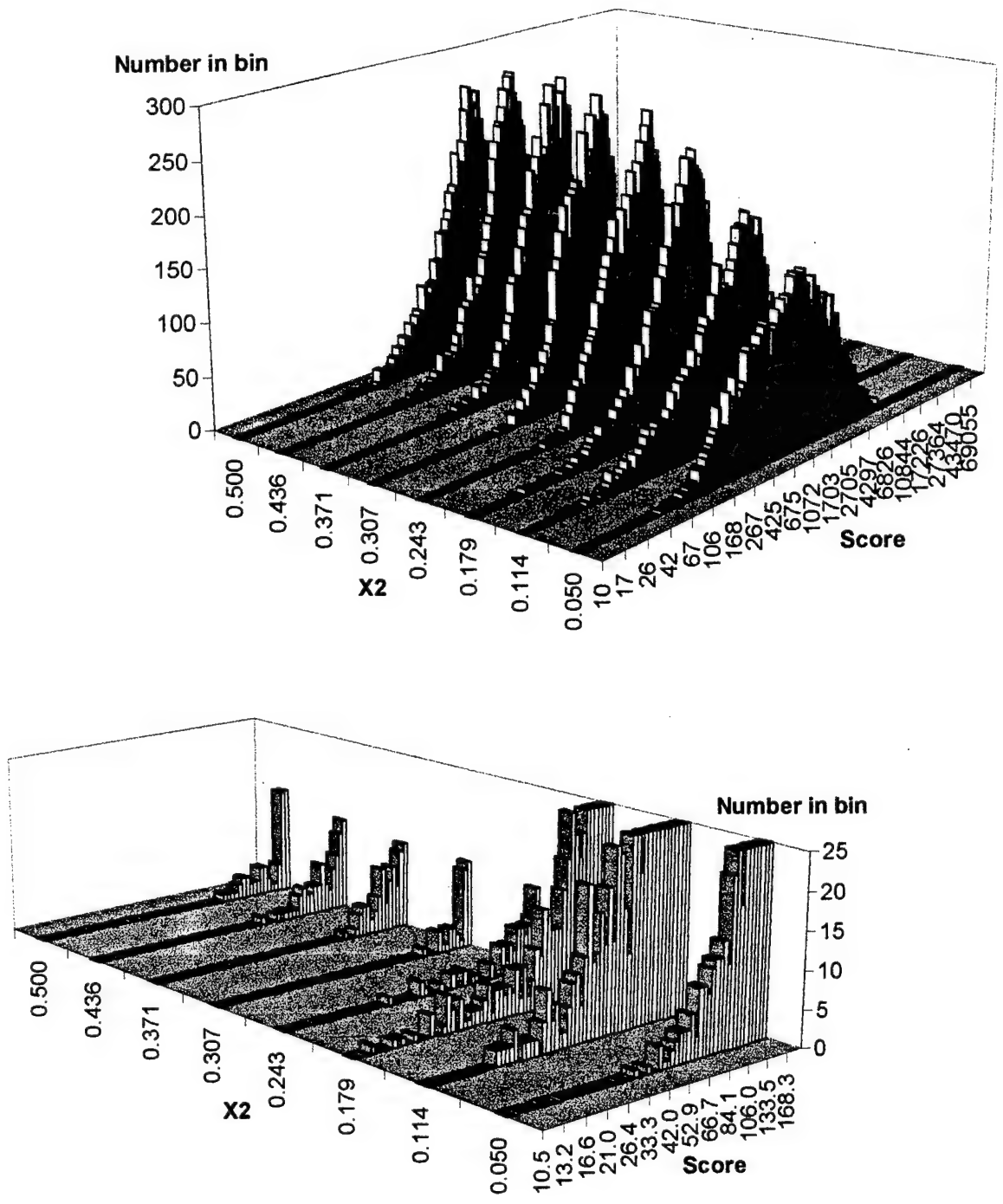


Figure 4.7. X2 Histogram and Close-up of low-score end.

X2 is a fairly sensitive parameter by itself, more so than the others. If X2 is larger than 0.243, the scores are limited to a significant degree. In addition, the scores are limited if X2 is too small. It seems that X2=0.179 is significantly better at producing low scores than any other value.

It appears that a larger standard deviation corresponds slightly to a possibility of better individuals, given that the other parameters are chosen correctly. There are more good individuals in the low-end tail of many of the distributions with the larger spread of scores, though they may not contain the best individuals. Interestingly, there are also more poor individuals in their distributions.

It is not too surprising that the distributions should be roughly normal about the reasonably bad score of about 1500. It requires careful design, or great deal of luck, to have something work well. It also requires a great deal of luck or skill to make something work extremely poorly. It requires very little luck, on the other hand, to have something that works fairly poorly. This is indeed the case with the loaded monopole.

Now that each variable has been discussed individually, it is of interest to see if the best scoring individuals found during the search are on the same hills, or on different ones. There is already some evidence of the latter, in the lack of good scores for X1=0.179. However, following is a list of all the antennas with scores less than 50 (corresponding to an average variation of 0.42dB from the mean).

Rank	Z1	Z2	Z3	Z4	X1	X2	Score
1	0.03	0.0433	0.0866	0.2009	0.1143	0.1786	13.0651
2	0.03	0.0733	0.1166	0.2309	0.05	0.1786	17.2837
3	0.0757	0.089	0.1923	0.2423	0.2429	0.1786	19.2299
4	0.2129	0.2262	0.3295	0.4437	0.2429	0.1786	23.5034
5	0.1214	0.1347	0.238	0.4166	0.3071	0.1143	26.4176
6	0.2586	0.2719	0.3452	0.588	0.3071	0.1143	26.4363
7	0.0757	0.089	0.1923	0.3066	0.5	0.1143	26.9067
8	0.1671	0.1804	0.2837	0.4623	0.3071	0.1143	27.0461
9	0.03	0.0433	0.0866	0.2009	0.2429	0.1786	28.6568
10	0.2586	0.3619	0.4052	0.4552	0.1143	0.1786	28.9299
11	0.0757	0.089	0.1923	0.3709	0.3071	0.1143	28.9446
12	0.2586	0.2719	0.3452	0.5237	0.3714	0.1143	29.6085
13	0.2586	0.2719	0.3152	0.4937	0.2429	0.1786	29.6956
14	0.1214	0.1347	0.238	0.3523	0.4357	0.1143	30.097
15	0.0757	0.089	0.1623	0.3409	0.3714	0.1143	31.2832
16	0.1671	0.1804	0.2837	0.398	0.4357	0.1143	32.3474
17	0.2129	0.2262	0.2995	0.478	0.3714	0.1143	33.5343
18	0.2586	0.3019	0.3152	0.6223	0.3071	0.1143	33.5724
19	0.0757	0.089	0.1623	0.3409	0.4357	0.1143	33.6865
20	0.1671	0.1804	0.2537	0.368	0.2429	0.1786	34.0604
21	0.2129	0.2262	0.3295	0.4437	0.4357	0.1143	35.8283
22	0.1214	0.1347	0.208	0.3866	0.3714	0.1143	36.6234
23	0.2586	0.2719	0.3452	0.5237	0.2429	0.1786	36.8269
24	0.3043	0.3176	0.3309	0.5737	0.2429	0.1786	36.8417
25	0.0757	0.089	0.1323	0.2466	0.05	0.1786	37.1446
26	0.03	0.0733	0.0866	0.2652	0.3714	0.1786	37.1656
27	0.2129	0.3162	0.3595	0.4095	0.1143	0.1786	37.5638
28	0.0757	0.089	0.1323	0.2466	0.2429	0.1786	37.9332
29	0.0757	0.089	0.1923	0.3066	0.4357	0.1143	38.4585
30	0.3043	0.3176	0.3309	0.638	0.3071	0.1143	38.8382
31	0.0757	0.089	0.1023	0.2809	0.3071	0.1786	38.8439
32	0.2129	0.2862	0.3295	0.3795	0.1143	0.1786	39.4351
33	0.1671	0.1804	0.2537	0.368	0.5	0.1143	41.8654
34	0.1671	0.1804	0.2537	0.4323	0.3714	0.1143	42.4109
35	0.1671	0.2104	0.2237	0.2737	0.05	0.2429	42.4896
36	0.2129	0.2562	0.3295	0.5723	0.3071	0.1143	42.6575
37	0.0757	0.089	0.1023	0.2809	0.3714	0.1786	42.7421

38	0.2586	0.3319	0.3752	0.4252	0.1143	0.1786	42.8912
39	0.3043	0.4076	0.4509	0.5009	0.1143	0.1786	43.0162
40	0.1671	0.2104	0.2237	0.4023	0.3071	0.1786	43.4411
41	0.2129	0.2262	0.2695	0.5123	0.3071	0.1143	44.8904
42	0.2586	0.3319	0.3452	0.588	0.2429	0.1786	46.5585
43	0.1214	0.1647	0.268	0.4466	0.3071	0.1143	48.4363
44	0.1214	0.1347	0.208	0.3223	0.5	0.1143	49.4246
45	0.2129	0.2262	0.2695	0.448	0.4357	0.1143	49.6259
46	0.1214	0.1347	0.178	0.2923	0.2429	0.1786	49.6631
47	0.03	0.0733	0.1466	0.3252	0.4357	0.1143	49.8516

Table 4.2. The top individuals with scores less than 50 from the exhaustive search

Looking at the table, it seems that there are similarities between 1, 2, 3, and 7. They could conceivably be on the same hill in the search space. However, 4, 5 and 6 are quite different from those four. They seem to have a completely different set of parameters. Thus, there are at least two hills of roughly equal magnitude in this search space. As one looks further down the table, it appears there are quite a few combinations that produce good results, though it should be noted that these are the top 47 individuals out of 65,536. Thus, it is not at all easy to achieve these results at random! And, with the complex interactions between variables that seem to exist, there are undoubtedly many more local minima than the two explored above, and it is very likely that they are not good. Note that, though the spacing of the data collected across the search space was completely even, the histograms are not at all even or smooth. Note also that, in the previous section regarding the optimization of the binary GA, the untuned GA converged to many sub-optimal results. This indicates the presence of local minima as well.

The likely presence of these poor local minima heightens the importance of the initial guess in a classical optimization. In addition, as mentioned above, the variables have a set of complex interrelationships with one another. Given any parameter value, the right combination of other parameters will make up a good antenna. Thus, the GA is a good choice of optimizer for this problem, not only because of its robustness and lack of need for an initial guess, but also because it accounts for the complex interdependencies of the variables and evolves all parameters at once.

A note before exploring the loaded monopole using a real-valued GA: it was not anticipated that the best answers would be asymmetric. All 47 of the good answers found in the exhaustive search, as well as those found by the binary GA, are asymmetric, though they were by no means constrained to be. This is quite a counter-intuitive result—it would seem that a symmetric, hemispherical pattern would be more likely to come from a symmetric antenna. Thus, if an engineer were to have to provide a classical optimization with a good initial guess, it seems very likely that the guess would have been symmetric, and the optimization would likely become stuck in a local minimum before converging to the better asymmetric solutions.

4.7 Results with real chromosome

Though the tuned binary GA is quite capable of solving this problem, it is of interest to compare its performance with that of a GA using real chromosomes. A run was performed with such a GA.

Mating occurred according to Adewuya's method [24], and a Gaussian mutation operator was used. There were 175 individuals, with 30% overlap, and 0.6% mutation rate. These parameters were discovered to be adequate while optimizing the Yagi antenna (described in Chapter 5), and so were used without modification to see if performance would be adequate.

The antenna that resulted, compared to the best binary individual, are listed in the table below:

	Real chromosome	Binary chromosome
Z1	0.0300 λ	0.0299 λ
Z2	0.0100 λ	0.0133 λ
Z3	0.0413 λ	0.0453 λ
Z4	0.110 λ	0.113 λ
X1	0.481 λ	0.413 λ
X2	0.142 λ	0.152 λ
Score	17.14	10.6

Table 4.3. Designs optimized by real and binary chromosomes

Note that the score achieved with one run of the real GA was better than all but one in the exhaustive search. Also, though the best binary GA score was better, the performance improvement was minimal: average variation was only 0.05dB better. In addition, to achieve this score required an experiment to determine the right parameters, and then it required 11 runs. The 11 runs themselves used 44,458 NEC2 simulations. The next closest score from these runs was 14.2.

The number of NEC2 runs needed for the real GA was 10,261. It achieved 99.9% of this score in 9769 NEC2 runs, and 99% in 7432 NEC2 runs. Of course, it also only required one GA run.

This seems to be a typical situation: the real GA takes more simulations than its binary counterpart, but is more certain of getting a good answer. It also seems more robust to its runtime parameters. The binary GA will often require tuning or re-running to be assured of adequate optimization. As will be seen in Chapter 5, the long-term savings in simulations can be significant.

4.8 Antenna validation and testing

As has been shown, the GA has produced many designs that have a nearly uniform power pattern over the entire hemisphere. One generated early in this research was chosen to be built and tested to validate the results. The dimensions for this loaded monopole are shown in Table 1, and a sketch of the antenna is shown in Figure 4.8.

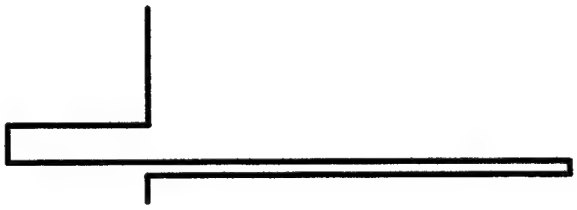


Figure 4.8. The loaded monopole

Name	Value (m)	Value (λ)
Z1	0.0056	0.0299
Z2	0.0024	0.0128
Z3	0.0079	0.0421
Z4	0.0236	0.1259
X1	0.0856	0.4565
X2	-0.0284	0.1515

Table 4.4. Dimensions of the loaded monopole

The score for this antenna was 30.9 for $\phi = 0°$, 45°, and 90°, and 13.7 for just the $\phi = 0°$ cut. This is among the best available.

The asymmetry of the resulting antenna would probably not have been produced analytically. It is hard to believe that this asymmetric structure produces near uniform coverage.

In the optimization processes described above, only limited information regarding the desired properties of the antenna were included in the cost functions for the GAs. Although hemispherical coverage was the goal, the cost function used by the GA was limited to three θ-plane cuts corresponding to ϕ angles of 0° (in the plane of the folded element), 45° and 90°, and a single frequency of 1600 MHz. Limiting the measures of quality in this way allowed the computations to proceed much more rapidly. However, after an optimal design is obtained, it is then necessary to conduct a more thorough analysis of the design. For the above designs, such an analysis was also completed using NEC2 and, as the results were satisfactory, the antennas were then fabricated and tested. If these results had not been satisfactory, it would have been necessary to either rerun the GA without modification, or include more properties in the fitness function and then rerun the GA.

The final computations were performed for intermediate angles and for frequencies from 1400 to 1800 MHz. The NEC2 output provided input impedance, current distribution and E_θ, E_ϕ and E_{Total} fields over the hemisphere.

Radiation patterns were computed for a set of E_θ, E_ϕ and E_{Total} (also denoted E_T) cuts in the θ-plane for 10° intervals in ϕ. The maximum difference between the maximum and minimum total fields for the whole hemisphere was less than 1.25 dB. An example of these results is shown in Figure 4.9 where the E_θ, E_ϕ and E_T fields are plotted for cuts in the θ-plane corresponding to azimuth angles of 0°, 45° and 90°, $\phi = 0°$ being a cut in the plane of the folded element while $\phi = 90°$ is a cut in the plane orthogonal to the folded element. Note that the $\phi = 0°$ cut has only an E_θ component whereas the $\phi = 45°$ and 90° cuts have both the E_θ and E_ϕ components. It has been found also that the total field is nearly uniform in all directions, as was the case for three ϕ angles shown. The computed input impedance was 133 + j229 ohms. (It was not attempted to optimize the impedance in this case—interest was primarily in the far-field directivity.)

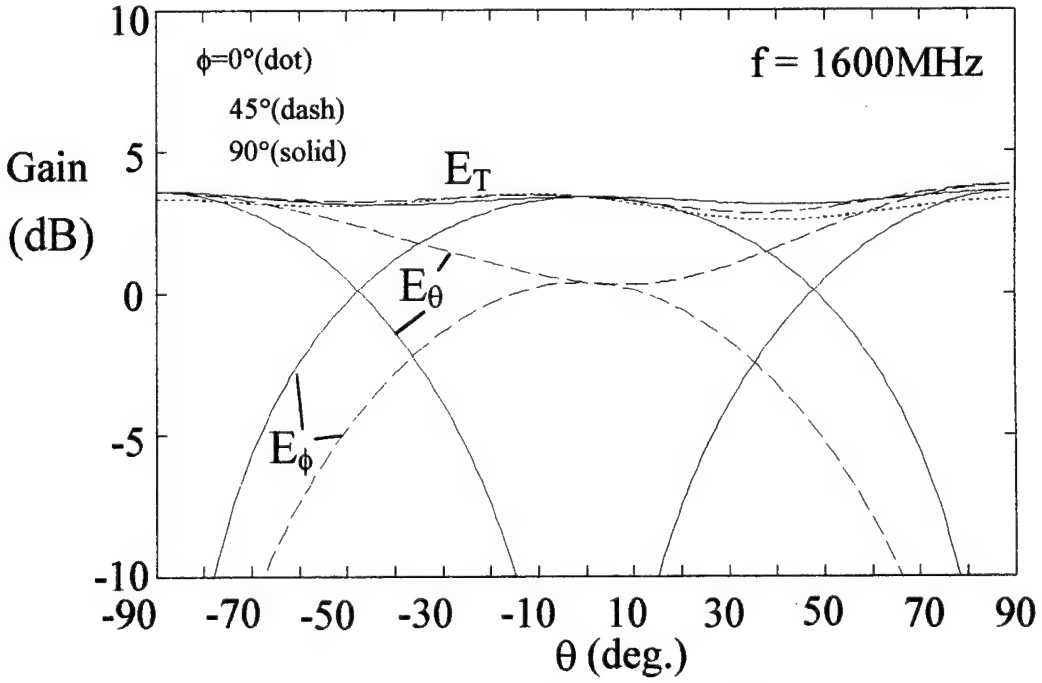


Figure 4.9. Computed E_θ, E_ϕ and E_T fields in θ-plane for ϕ=0°, 45° and 90° at 1.6 GHz

The frequency dependence of the loaded monopole was examined. The E_θ, E_ϕ and E_T patterns were computed for θ-plane cuts corresponding to $\phi = 0°$, 45°, and 90° in increments of 50 MHz over the range from 1400 to 1800 MHz. In Figure 4.10 the computed results are plotted for $\phi = 45°$ for increments of 100 MHz, and are representative of results at other ϕ angles. It is seen that the maximum variation in power gain over the hemisphere over a 25% frequency range is only about 6 dB.

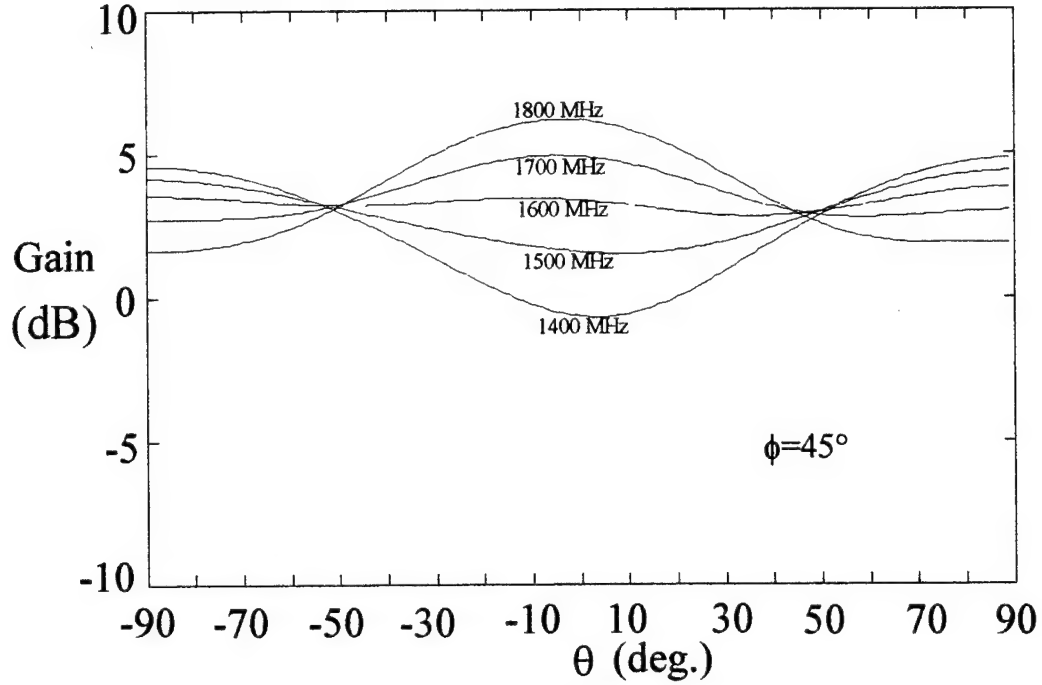


Figure 4.10. Computed E_T field in θ-plane for ϕ=45° at frequencies from 1.4 to 1.8 GHz.

The optimal antenna design was fabricated and tested. The antenna was made from 1.8 mm diameter wire, and the dimensions of the test antenna approximated those of the computed antenna to about +/- 1 mm. The loaded monopole was mounted over a 1.2m x 1.2m ground plane and fed from a coaxial line. E_θ and E_ϕ radiation patterns were measured in an anechoic chamber. The total E-field

$$E_T = \sqrt{(E_\theta^2 + E_\phi^2)}$$

was then calculated. Since primary interest was in the directional properties of the antenna, instead of efficiency, only relative gain (i.e., directivity) was measured, and a measurement of its absolute gain was not attempted. (Measuring absolute gain, which includes a measure of antenna efficiency, requires one to calibrate the measuring system, including the sending antenna, and requires one to factor in impedance mismatch between the antenna under test and the measurement equipment. Since this kind of accuracy was not needed to validate the NEC2 simulation, these calibrations were not performed.)

The E_θ, E_ϕ and E_T components are shown in Figure 4.11 for the $\phi = 45°$ θ-plane cut. The computed and measured patterns are similar except for the ripples and loss of signal near the horizon; these effects are due to the finite ground plane. This ripple will be discussed in more detail in Chapter 6, Section 6.4.1. Computed and measured patterns for ϕ = 0° and 90° cuts were also in good agreement. The measured total field patterns are shown in Figure 4.12 for cuts in the θ-plane corresponding to angles of ϕ=0°, 45° and 90°. Note that the total field varies by less than 4 dB over nearly the entire hemisphere.

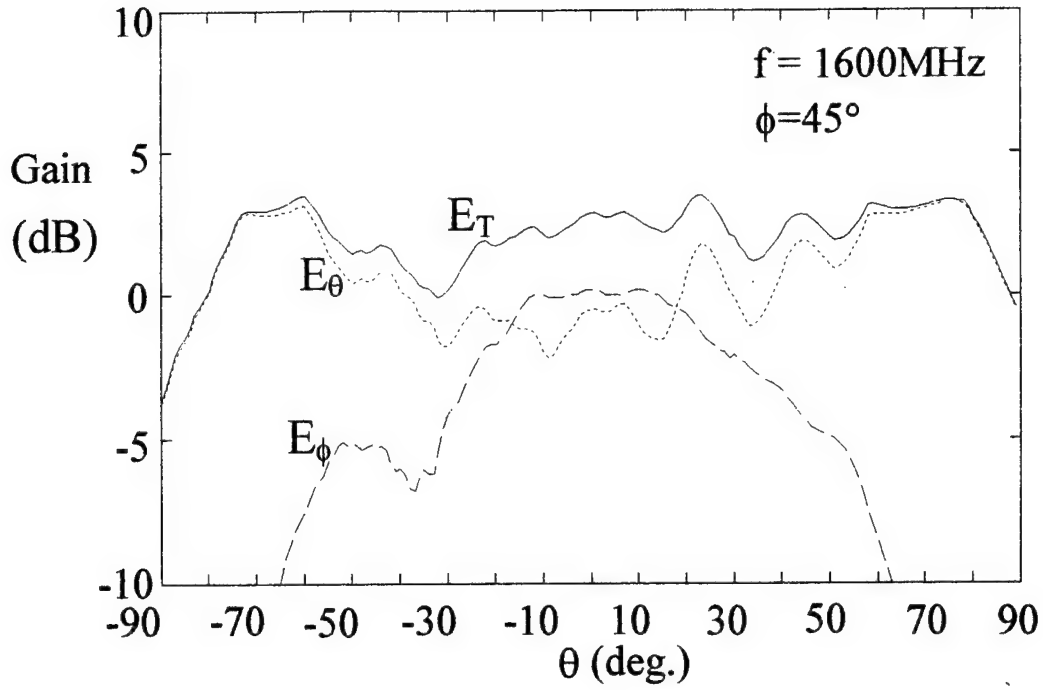


Figure 4.11. Measured E_θ, E_ϕ and E_T fields in θ-plane for ϕ=45° at 1.6 GHz.

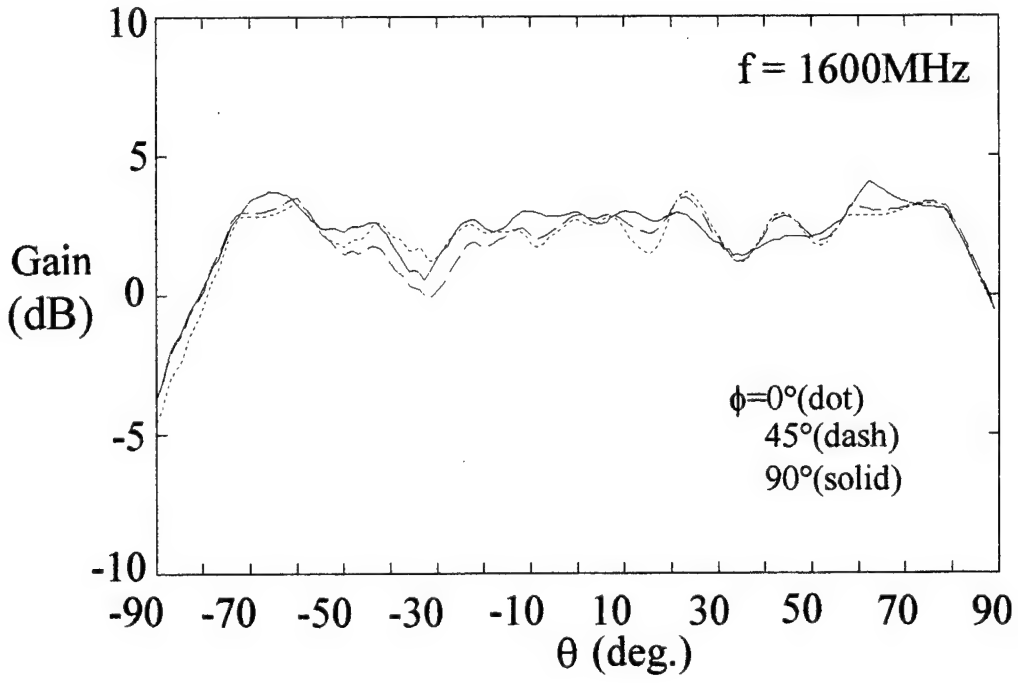


Figure 4.12. Measured E_T field in θ-plane for ϕ=0°, 45° and 90° at 1.6 GHz.

The E_θ and E_ϕ fields were also measured over the range from 1400 to 1800 MHz and the corresponding E_T field was calculated for comparison with the computational results of Figure 4.9. In Figure 4.13, E_T is plotted for $\phi = 45°$ for increments of 100 MHz over this range. The maximum variation in power over the hemisphere (except for very low

elevation angles) was about 6 dB, as it was in the computed results. The ripples in the pattern are due in large part to the finite ground plane, as will be discussed in greater detail in Chapter 6.

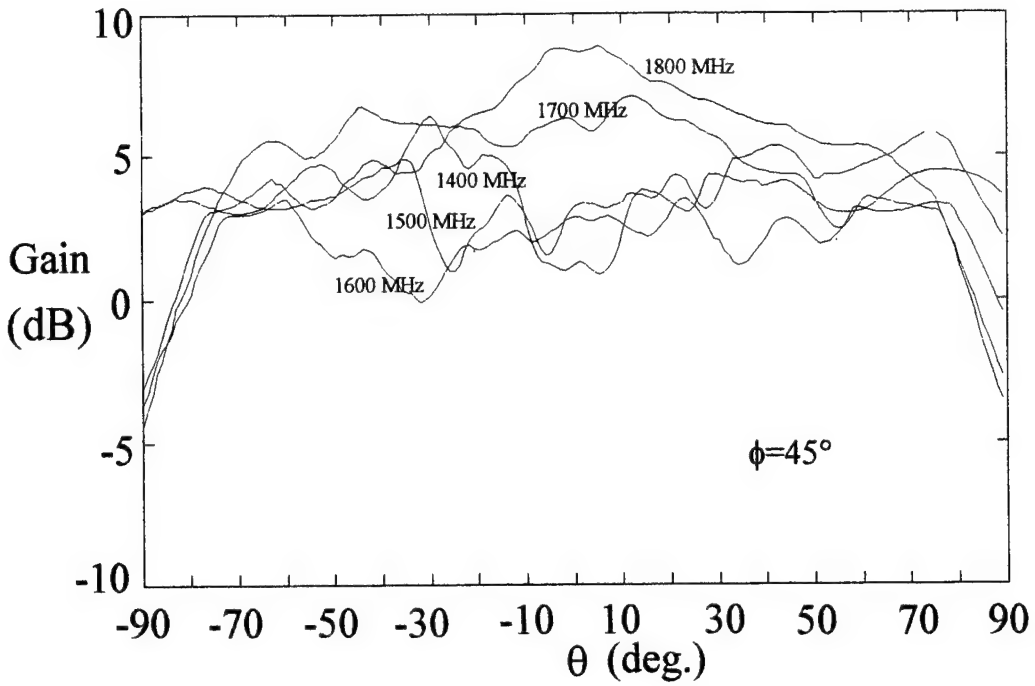


Figure 4.13. Measured E_T field in θ-plane for ϕ=45° at frequencies from 1.4 to1.8GHz.

4.9 Conclusion

This chapter dealt with a very interesting antenna: the loaded monopole. It was optimized with a GA, an experiment was run to determine the best parameters for the GA, and a design was built and tested. It shows both the power of the GA to search counter-intuitive search spaces and find excellent, though unusual, results, and the robustness of the GA to initial parameters. Even an untuned binary GA worked well on occasion.

The next chapter discusses another classical design: the Yagi antenna, and its optimization for a traditional purpose—high gain—and a non-traditional purpose: an Arecibo feed antenna. It will explore further the marriage of antenna design and GA optimization methods, expanding into multi-goal optimization, and a more extensive comparison between the real and binary GA.

Chapter 5: The Yagi antenna

5.1 Introduction

This chapter will look at the application of the GA to the optimization of the Yagi antenna. Unlike the loaded monopole, the Yagi antenna has been optimized by the GA for several different applications, all of which have more than one characteristic to be optimized. Thus, in this chapter, not only will a new antenna configuration be explored, but the application of the GA to designs with more than one goal will be investigated as well.

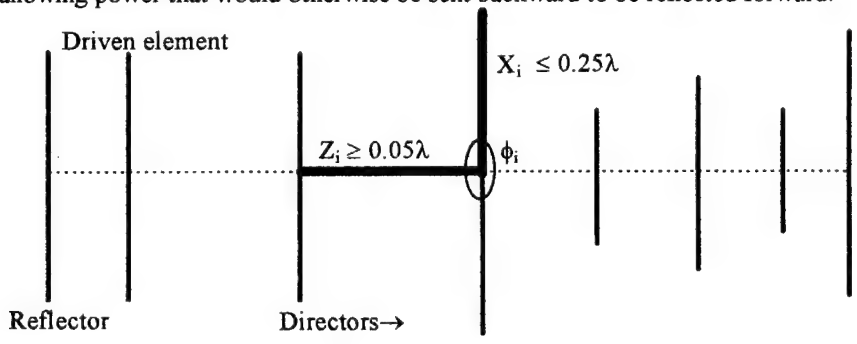
The Yagi antenna evolved as a special configuration of an endfire array. It is a traveling wave antenna with a surface wave that propagates along the array with a phase velocity slightly less than that of free space. It was first proposed by Prof. Yagi and his student, S. Uda, in the late 1920s. It consists of a single driven element and a number of parasitic elements made up of a reflector and a set of directors. The Yagi has been exhaustively investigated, both theoretically and experimentally, for many years. The Yagi configuration has not been amenable to theoretical analysis since it is an array of elements of different lengths with non-uniform spacing and thus cannot be treated using conventional array theory. Most mathematical analyses have been restricted to relatively short arrays; progress throughout the years for longer arrays has been achieved mostly experimentally and computationally.

Over the years the performance of Yagi antennas has been slow to improve. It is believed that maximum gain is achieved by controlling the phase velocity of the surface wave—the Yagi structure must be designed so that the surface wave is properly retarded. This has been accomplished with some success by logarithmically tapering the elements; the director spacings are gradually increased while the lengths are gradually decreased until they approach constant values at a distance of about 3 or 4 wavelengths from the driven element. Minor changes in the antenna configuration have produced only a small improvement in performance.

The Yagi is light weight and inexpensive and it has been widely used for many high gain and narrow frequency band applications, particularly by the amateur radio community. It does not appear to have been previously used for low sidelobe and wide bandwidth applications, but it will be optimized for such an application in this chapter.

5.2 The search space

As shown in the figure below, the Yagi antenna is a series of parallel wires. One element is driven, one element is behind the driven element and is called the reflector, and all other elements are called directors. The highest gain can be achieved along the axis and on the side with the directors, as would be expected. The reflector acts like a small ground plane, allowing power that would otherwise be sent backward to be reflected forward.



For N wires:
(2N - 1) genes for geometry (or 3N-1 if ϕ included)
+ 1 gene for wire diameter = 2N real genes

Figure 5.1. Yagi Antenna

The conventional Yagi design includes geometry variables of length for each element, spacing between elements, and the diameter of the wire. Thus, with N elements, there are N length variables, N-1 spacing variables, and one wire diameter variable, giving 2N variables total. The driven element is driven from its center.

There are a couple of different ways to encode the spacings into the chromosome. The simplest way is to allow the maximum gene value to have the greatest allowable spacing, and the minimum gene allele to have the minimum spacing. This is only good if the required boomlength is unknown, however, for it will constantly be changing using this encoding. It is known that the boomlength affects the maximum gain possible with the Yagi antenna, thus a second method should be used if a boomlength is to be specified by the engineer.

This second method takes all spacing gene alleles and sums them to give a total allele value. This value is then divided into the boomlength, to give a unit spacing. The alleles of each spacing gene is then multiplied by this unit spacing to give the actual spacing for each element. In this way, the whole boomlength is used, and the spacings tend to be more even. A certain minimum spacing is maintained between all elements, however, to keep them from overlapping.

One other variable beyond that of the conventional Yagi that produced good results is shown in the above figure. It is designated as ϕ_i. This is a degree of freedom relating to the rotation of the element out of the plane of the antenna. The reason to allow this extra variable is discussed in the last section of this chapter. Other variables were tried with no successful results, but will also be mentioned in that section.

5.3 Optimization for VSWR and gain only

To begin, it was decided to optimize the gain of four Yagi antennas having boom lengths ranging from 3.6 to 6.1 λ at 432 MHz; VSWR was of secondary importance and back and sidelobes were not included in the optimization yet. The reason VSWR was included from the beginning was because the optimization would produce antennas with extraordinarily low impedances, and these structures were believed to be in supergain, and thus not actually achievable.

The gains, radiation patterns and VSWRs of these antennas were computed using NEC2 and then compared with the gains achievable using currently available design techniques [26, 27]. A conventional Yagi and a GA-optimized, or genetic, Yagi, each having the same boom length, were then fabricated and tested.

For this optimization, both real and binary chromosomes were used. For both chromosome types, a steady-state GA was used, with a fitness-weighted virtual roulette wheel, and Gaussian mutation for real chromosomes, with a standard deviation of 10% of the gene range.
The antennas were optimized for gain and VSWR at a frequency of 432 MHz. The cost function was
$F = -G + C_1 \times (VSWR)$
where G is the endfire gain and C_1 is 1 when the VSWR is greater than 3.0 and 0.1 when the VSWR is less than 3.0. The objective was to minimize F.
Each element has a range of lengths up to 0.75λ with minimum spacings of 0.05 λ. The antenna was constrained to use the entire boom length. The possible wire diameters range from 2 to 6 mm with 1 mm increments. Real chromosomes were used to optimize the Yagi with a boom length of 5.16 λ and binary chromosomes were used for the other Yagis. The binary GA had the following parameters: population 50, overlap 30%, mutation rate 2%, mated with two-point crossover. This was an unusual set of parameters, but because during preliminary GA runs with these antennas good solutions were so elusive, it seemed best to use small populations that would converge to a useful solution that could be hillclimbed in less than 1600 NEC2 simulations. In limiting these simulations per GA run, many such runs could be performed in search of the elusive design that showed superior performance. For the real GA, a population of 175 Yagi configurations was used, with a 30% overlap and a 0.6% mutation rate. Adewuya's method [24] was used for mating, and Gaussian mutation was employed. The GA using the real-valued chromosomes appeared to converge more quickly.

5.3.1 Results

Configurations with excellent properties were found for the 14, 17, 17 and 22 element Yagis having corresponding boom lengths of 3.60 λ, 4.88 λ, 5.16 λ and 6.10 λ. After optimal configurations were produced by the algorithm,

conventional Yagis having corresponding boom lengths were selected for comparison, with the exception of the Yagi having a boom length of 4.88 λ. This particular Yagi was optimized to have 18 elements and a boom length of 5.16 λ, however, the end element of this genetic Yagi was very small and had a very low current; it was found that removing it from the antenna did not change any of the characteristics.

The GA produced configurations that were quite different from a typical Yagi. The director lengths and spacings for the conventional and genetic Yagis are shown in Table 1. Note that the conventional Yagis have directors with lengths that gradually decrease and spacings that gradually increase along the array. For the genetic Yagis, neither the lengths nor the spacings showed any systematic change along the antenna; they appeared for all practical purposes to vary at random.

	CONVENTIONAL YAGI					
BL	3.60λ		5.16λ		6.10λ	
El.	L	S	L	S	L	S
	cm	cm	cm	cm	cm	cm
1	34.5		33.7		34.6	
2	32.8	13.0	32.5	12.7	34.0	10.4
3	30.8	5.5	30.0	5.7	32.1	4.2
4	30.4	12.5	29.8	11.1	31.1	7.8
5	30.0	15.0	29.5	14.9	30.5	10.8
6	29.6	17.5	29.1	17.5	30.1	13.4
7	29.3	19.5	28.9	19.4	29.7	15.6
8	29.3	21.0	28.7	20.8	29.5	17.6
9	28.9	22.0	28.6	21.9	29.3	19.2
10	28.9	23.0	28.4	22.7	29.1	20.6
11	28.9	24.0	28.3	24.0	28.9	21.8
12	28.5	25.0	28.1	24.9	28.8	22.8
13	28.5	26.0	27.9	25.7	28.6	23.7
14	28.1	26.0	27.8	26.4	28.5	24.3
15			27.5	26.7	28.4	25.1
16			27.3	27.4	28.3	25.6
17			27.1	27.6	28.1	26.1
18			27.0	27.8	28.0	26.4
19					27.9	26.8
20					27.8	27.1
21					27.7	27.4
22					27.6	27.5

	GENETIC YAGI							
3.60λ		4.88λ		5.16λ		6.10λ	
L	S	L	S	L	S	L	S
cm	cm	cm	cm	cm	cm	cm	cm
33.1		33.1		33.8		33.6	
34.6	17.5	30.4	13.4	31.5	13.2	28.8	9.3
31.5	10.1	30.9	9.7	30.3	17.3	31.5	2.1
30.4	20.5	29.8	18.2	29.5	24.2	30.4	15.5
17.9	10.1	11.4	14.4	27.9	21.6	29.3	25.6
22.2	12.6	29.3	7.8	27.6	12.6	11.9	3.0
29.3	4.6	14.1	24.8	27.2	18.3	20.6	5.9
28.8	32.0	28.2	4.9	27.2	13.7	28.8	19.8
28.8	29.0	20.6	3.2	28.0	23.3	28.2	31.3
26.0	26.0	27.7	26.5	25.8	21.2	28.8	31.3
27.1	6.1	27.7	34.3	27.3	15.0	28.8	22.2
28.8	28.0	28.2	24.8	27.7	27.4	12.5	15.0
28.8	29.5	25.5	38.1	27.6	29.7	27.1	25.5
29.3	28.0	27.7	37.1	27.4	33.2	27.7	29.9
		27.7	25.8	27.7	29.4	27.7	28.0
		28.2	32.4	27.9	32.4	22.2	26.5
		29.3	22.9	28.6	25.9	26.0	16.5
						27.1	27.0
						19.5	11.2
						27.7	20.3
						28.2	30.9
						28.8	24.0

Table 5.1. Dimensions of high-gain Yagi antennas

The gains and VSWRs were computed for both designs using NEC2. Although bandwidth was not included in the optimization of the genetic Yagi, the frequency dependence of the antennas was computed for completeness. Table 2 shows the computed gains and VSWRs for both types of antennas near the design frequency of 432 MHz. The conventional Yagis have nearly uniform gain and relatively low VSWRs over most of the band. The genetic Yagis have a higher gain at 432 MHz, but the gain drops off sharply near 440 MHz, particularly for the 3.6 and 5.16 λ antennas. Also, the VSWRs are low at 432 MHz but tend to increase sharply at frequencies other than 432 MHz. The computed E- and H- plane radiation patterns are shown in Figure 5.2 for both types of antennas with a boom length of 5.16 λ at the design frequency of 432 MHz. The sidelobes and backlobe for the conventional Yagi are about 15 and 22 dB down, but those for the genetic Yagi are down 16 and 13 dB respectively. The E-plane patterns for the other Yagis are shown in Figures 5.3, 5.4 and 5.5. Once again the sidelobes for the conventional and genetic Yagis are about the same, however the backlobes for the conventional Yagis are lower.

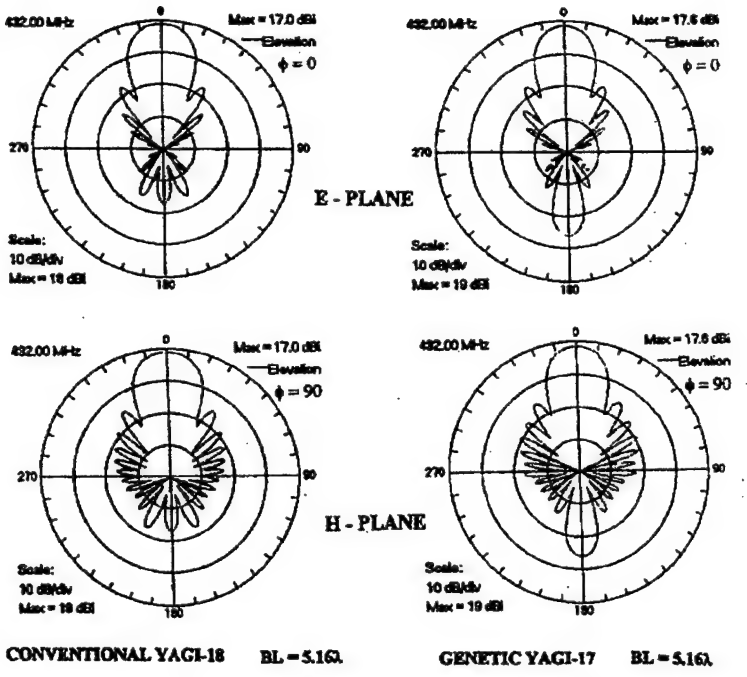


Figure 5.2. Computed E- and H- plane patterns of 5.16 λ boom length conventional and genetic Yagis at 432 MHz.

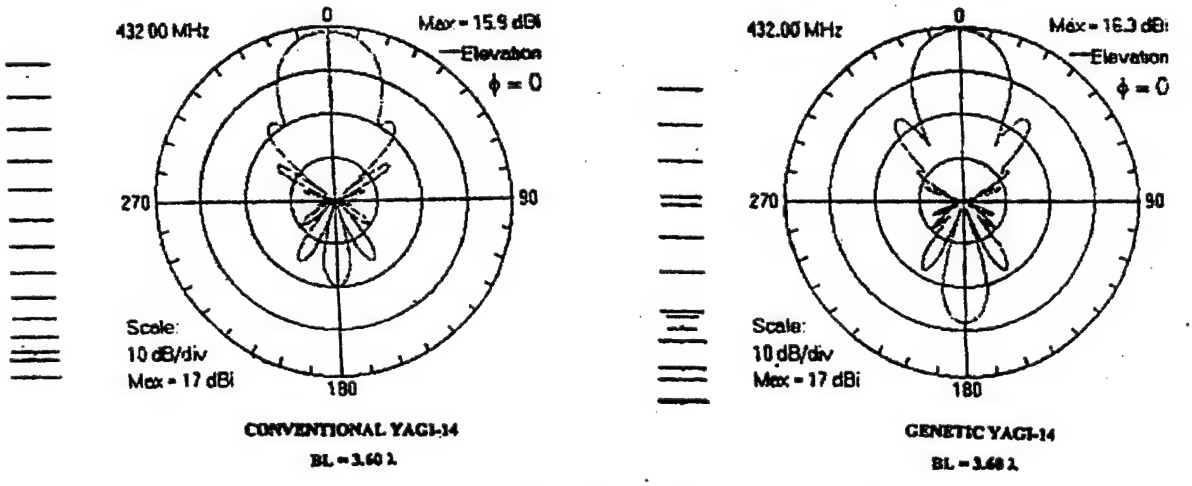


Figure 5.3. Computed E- plane patterns of 3.60 λ boom length conventional and genetic Yagis at 432 MHz.

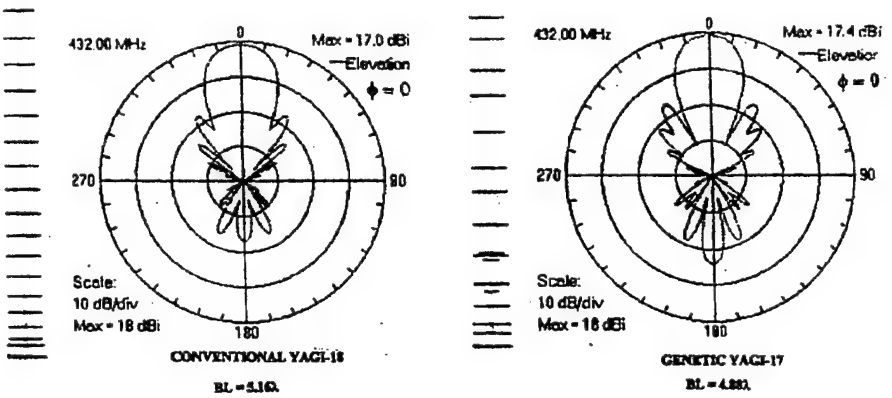


Figure 5.4. Computed E- plane patterns of 5.16 λ boom length conventional and 4.88 λ genetic Yagi at 432 MHz.

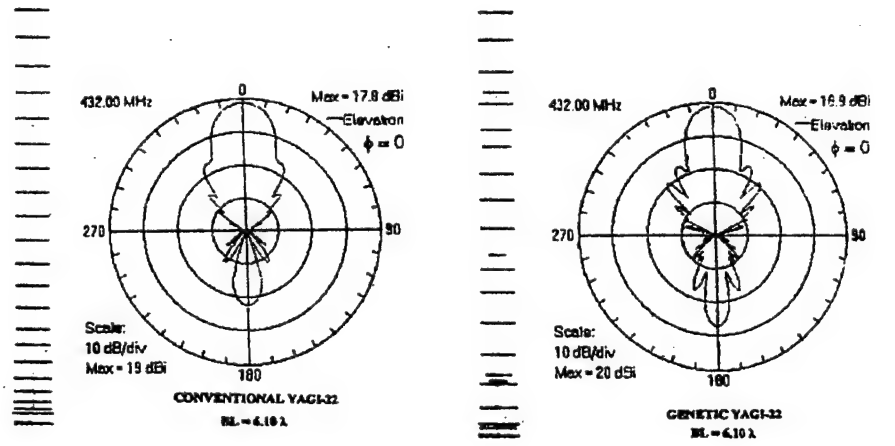


Figure 5.5. Computed E-plane patterns of 6.10 λ boom length conventional and genetic Yagis at 432 MHz.

CONVENTIONAL YAGI					GENETIC YAGI				
N	BL λ	FREQ MHz	GAIN dBi	VSWR	N	BL λ	FREQ MHz	GAIN dBi	VSWR
14	3.60	424	15.5	1.41	14	3.60	424	15.4	1.88
		428	15.8	1.11			428	16.0	1.80
		432	15.9	1.23			432	16.3	1.09
		436	15.7	1.60			436	16.1	9.5
		440	15.5	1.85			440	9.4	39
					17	4.88	420	15.8	5.10
							424	16.4	4.50
							428	17.0	3.70
							432	17.4	1.55
							436	17.0	13.0
18	5.16	424	16.5	1.37	17	5.16	420	16.4	2.55
		428	16.8	1.13			424	16.9	2.60
		432	17.0	1.03			428	17.3	2.35
		436	17.1	1.27			432	17.6	2.80
		440	17.1	1.43			436	16.5	16.0
							440	6.4	18.0
22	6.10	424	17.8	1.34	22	6.10	416	16.4	23
		428	17.8	2.28			420	17.1	19
		432	17.8	2.92			424	17.9	12
		436	17.5	2.81			428	18.5	9.3
		440	17.5	5.20			432	18.9	1.51

Table 5.2 Summary of Yagi computations

The conventional and genetic Yagis having a boom length of 5.16 λ were fabricated, and their gains, E- and H-plane patterns and VSWRs were measured. Since the antenna pattern range did not operate satisfactorily at frequencies below about 800 MHz and since the full scale antennas would have been quite large it was decided to work with a ¼ scale model having a center frequency of 1728 MHz. The conventional Yagi elements were made of 1.2 mm (3/64 in.) copper rod while the genetic Yagi used 1.6 mm (1/16 in.) rod. These elements were inserted into ½ in. PVC pipe. It is estimated that the Yagis were built to an accuracy of about +/- 0.5 mm. The measurements were made over the frequency range from 1650 to 1750 MHz which corresponded to the full scale range of about 412 to 438 MHz.

Since this antenna's quality is particularly sensitive to measurement technique, the measurement equipment and method used will now be described in detail. The VSWRs of the conventional and genetic ¼-scale Yagis, each having a boom length of 5.16 λ, were measured with a Hewlett-Packard 8510 Network Analyzer over the frequency range from 1650 to 1750 MHz. The gain and radiation pattern were measured on a 2600 ft far field range using a Flam & Russell Model 959 Automated Measurement Workstation with a Hewlett Packard 8530A receiver. The transmitter was a Hewlett Packard 8340B synthesized source used in conjunction with a 4- foot parabolic dish which provided an ample signal level and dynamic range. A Scientific Atlanta model 12-1.1 standard gain horn was used as a reference antenna. There are two main sources of error in the measurement of absolute gain: the first is the uncertainty of the true gain of the standard gain horn; this has a 1 σ value of +/- 0.17 dB. The second source of error arises from reflections and multipath from the terrain. Fluctuations of approximately +/- 0.3 dB were observed as the antenna was moved back and forth along a track. Thus, there is an uncertainty of about +/- 0.5 dB in the measurement of the absolute gain and about +/- 0.3 dB in the relative gain. In addition, it is necessary to correct for the mismatches that may arise from the transmission line to both the standard gain horn and the Yagi antennas. There are two options that can be used for this correction: a matching device (e.g. stub tuner) can be inserted to reduce the mismatch losses, or the input VSWR can be measured and the corresponding loss in gain can then be calculated as follows,

$$\text{Gain loss (dB)} = 10 \log_{10} \{1 - [(\text{VSWR} - 1)/(\text{VSWR} + 1)]^2\}$$

A combination of both methods was used. Since the Flam & Russell system automatically swept over the frequency range, it was not practical to match the antennas at all frequencies. The standard gain horn had VSWRs below 1.2 so the corresponding losses were less than .04 dB. The conventional Yagi had a maximum VSWR of 2.1, so corrections

of up to about 0.6 dB were needed. For the genetic Yagi, the VSWRs were between 2.5 and 3.0 over most of the band, but increased sharply above 1730 MHz. At three frequencies in the vicinity where the Yagis exhibited maximum gain a double stub tuner was used to minimize the mismatch loss. In Figure 5.6 is a plot of the corrected gains for the conventional and genetic Yagis as a function of frequency. Note that the conventional Yagi has a maximum gain of 16.2 dB at 1700 MHz—it was slightly lower at 1728 MHz, the scale frequency that corresponds to the full scale frequency of 432 MHz. The genetic Yagi had a maximum gain of 17.0 dB at 1730 MHz. As expected, the gain of the conventional Yagi was reasonably flat over most of the band, whereas the gain of the genetic Yagi decreased sharply at the higher frequencies. The measured gains were somewhat lower than the computed gains. As mentioned previously, this can be attributed to the uncertainty in the gain of the Standard Gain Horn. However the relative gains of the Yagi antennas are in good agreement with the computations.

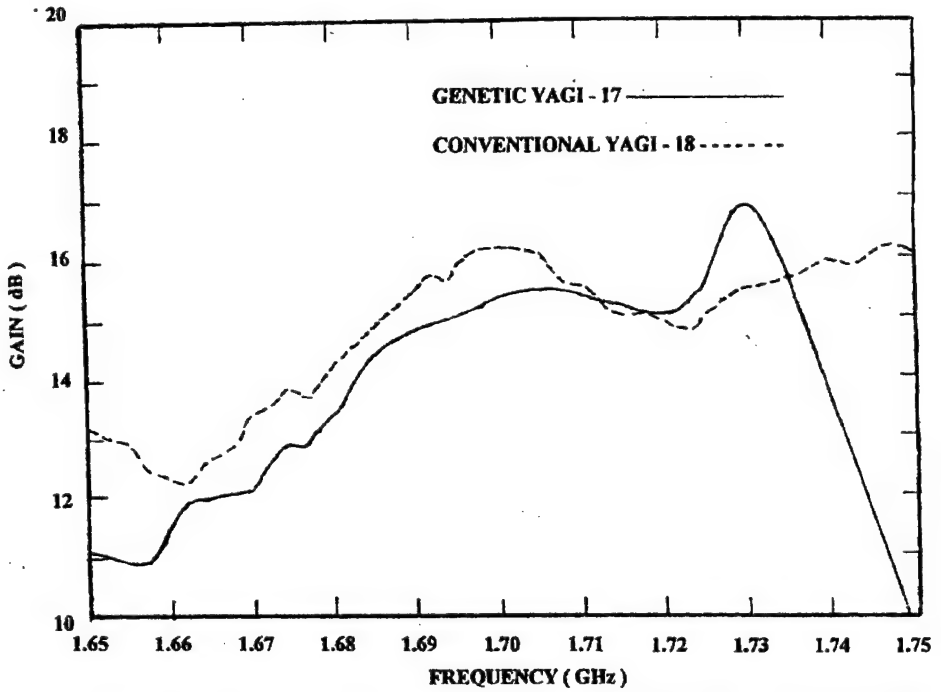


Figure 5.6. Measured gains of 5.16 λ boom length conventional and genetic Yagis from 1.65-1.75 GHz. Solid line is genetic Yagi, dotted line is conventional Yagi.

5.3.2 Conclusion

It has been shown computationally and verified experimentally that by using a GA in conjunction with NEC2 it is possible to synthesize, at a single frequency, a Yagi antenna design that has a significantly higher gain than that of a state-of-the-art Yagi. This case focused on maximum gain and did not weight the VSWR very heavily. Other parameters like bandwidth and sidelobe and backlobe levels could have been included in the cost function, but it would be expected that at the price of some loss of gain, as will be seen to occur in the next section.

5.4 Optimization for gain, backlobe level and sidelobe level

While success was achieved with gain and VSWR only, the Yagi antenna has another important performance characteristic that has been ignored: sidelobes. One particularly important sidelobe occurs directly opposite the main beam, called, for obvious reasons, the backlobe. By limiting the sidelobes, one will increase the signal to noise ratio for the antenna, for it will gather less energy from surrounding sources. Limiting the backlobe is particularly important, for often the antenna is pointed skyward, and the backlobe is therefore looking at the ground. The earth itself, as a warm body, is a source of noise, and thus a large backlobe can increase the noise floor considerably.

Two terms are added to the cost function to include both sidelobes and backlobes. The cost function is now:

$$F = -C_0 \cdot G + C_1 \cdot (\text{VSWR}) + \text{SLL term} + \text{BLL term}.$$

$$\text{SLL term} = \begin{cases} C_2 \cdot (\text{SLL}-\text{DSLL})^2 & \text{while SLL} < \text{DSLL} \\ C_2 \cdot (\text{DSLL}-\text{SLL}) & \text{while SLL} > \text{DSLL} \end{cases}$$

$$\text{BLL term} = \begin{cases} C_3 \cdot (\text{BLL}-\text{DBLL})^2 & \text{while BLL} < \text{DBLL} \\ C_3 \cdot (\text{DBLL}-\text{BLL}) & \text{while BLL} > \text{DBLL} \end{cases}$$

where G is the endfire gain and C_1 is 1 when the VSWR is greater than 3.0 and 0.1 when the VSWR is less than 3.0, as before. C_0 is a constant that allows control over the relative importance of the gain compared to the other variables. It is generally set to 1. SLL is the largest sidelobe level (SLL) in the pattern compared to the maximum gain of the main beam, and the details of its measurement will be discussed shortly. DSLL is the Desired SLL. C_2 is similar to C_1 in that it allows us to designate the relative importance of the sidelobe level in the equation. It is usually set between 1 and 0.3 if sidelobe level is included. The term for the Backlobe Level (BLL) is similar to the SLL term. C3 can vary from 1 to 0.3 as well. The objective, as before, is to minimize F. Note that both the SLL and the BLL terms are quadratic while they are sub-standard, but they become linear once the design objective has been reached, so that they have the most impact on the cost function while they require the most attention. As the engineer will be somewhat indifferent once design objectives have been reached, the impact of these factors is reduced, but not eliminated, as a better SLL or BLL is always desired, at least somewhat.

Since BLL is measured 180° from the center of the main lobe, i.e., where θ=180°, it is easy to determine. However, SLL is not so straightforward. It is necessary to use the highest lobe in the pattern other than the main beam for the SLL, so all gains outside the main beam nulls are checked. The greatest of these is considered to be the SLL, and can be the same as the BLL should the backlobe also be the worst sidelobe. Hence it is important for the user to specify the beamwidth for the GA as being from null-to-null as opposed to the 3dB points, as is more common. Otherwise, the SLL can be erroneously given by part of the main lobe outside the 3dB points.

Computed results with binary and real chromosomes

Using a boom length of 5.16λ, and 18 elements, a binary GA and a real GA were used to optimize a design for gain, SLL and BLL, and their performance was compared.

The best design for the binary GA resulted from the following parameters: population 150, 30% overlap, 0.6% mutation rate, two-point crossover, and 6 bits/variable. C1 was 1, C2 was 0.15, and C3 was 0.25. The GA was also modified to include a parameter to keep the GA from running too long. A maximum number of 4,000 NEC2 simulations was specified. After these 4,000 NEC2 runs, a stochastic hillclimber finished the optimization. It was determined that after 4,000 simulations, the GA frequently had arrived on the hill it would spend the rest of its time exploiting. However, as discussed in Chapter 3, once a GA has converged to a single hill, it is often of use to change strategies and exploit the hill with another optimization method. Though a more classical optimization approach could have been used, a stochastic hillclimber was found to be adequate for exploiting these hills.

This hillclimber randomly searched an area, changing one variable at a time by one step up or down, to see if the individual improved. If it did, the hillclimber kept this new individual and modified another variable. It stopped when

it had stepped each variable up and down and found no better scores—i.e., it had become stuck. It was found that using this hillclimber saved NEC2 simulations by exploiting a hill more rapidly than could the nearly-converged GA.

In this case, the GA converged to an individual with a score of -0.936. The hillclimber took this individual as its starting point and optimized it until it had a score of -16.54! While the GA took 4,078 NEC2 simulations (the GA is permitted to finish a generation once it reaches the 4,000 mark), the hillclimber took 2,046 simulations—a typical number of runs to use.

This optimized design has a gain of 16.63dB, with a backlobe 25.0dB down from the maximum gain, and a sidelobe level of 18.2dB down from the maximum gain. This antenna's parameters are shown in the following table, and its gain pattern along with the conventional 18-element Yagi are shown in the following figure. It had a score of -16.54. The cost of finding this antenna in NEC2 simulations and GA runs is shown in Table 5.3.

Length (λ)	Separation (λ)
0.4608	0.0000
0.4844	0.4650
0.4608	0.5346
0.4297	0.8018
0.4142	1.0690
0.4064	1.2334
0.3906	1.4692
0.4064	1.7363
0.3983	1.9721
0.1719	2.0573
0.3906	2.1190
0.3906	2.4811
0.3750	2.9937
0.3983	3.4664
0.3983	3.8524
0.3983	4.2778
0.3750	4.7745
0.3828	5.1603
Wire dia:	0.00720λ

Table 5.3. Binary GA Antenna dimensions

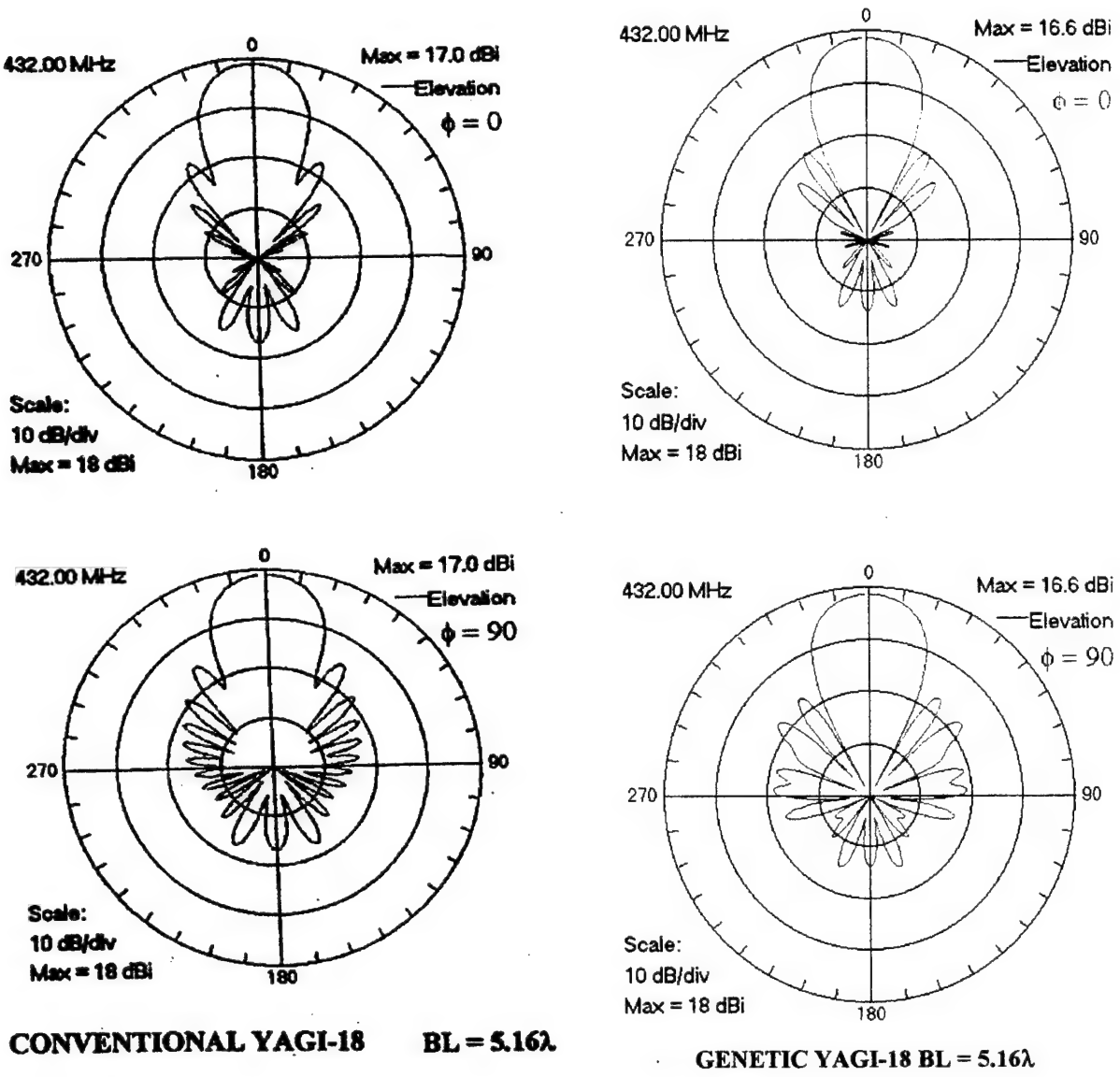
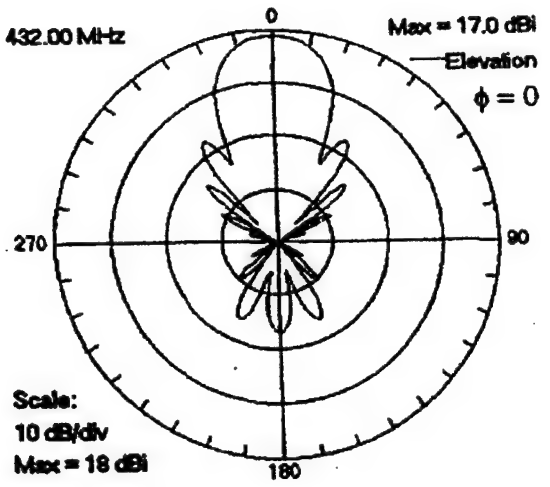


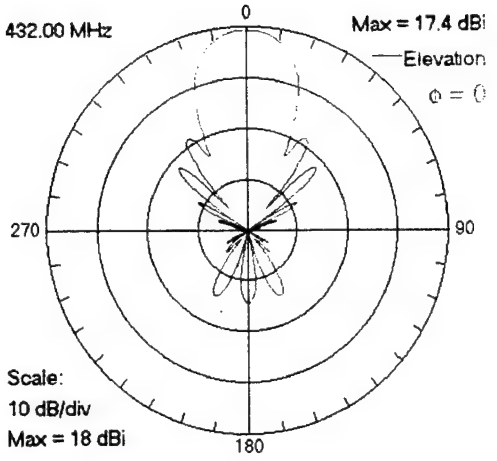
Figure 5.7. Binary GA Antenna gain pattern vs. conventional Yagi

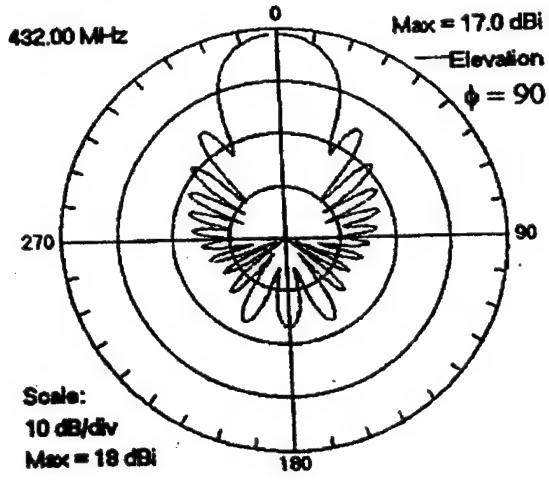
The GA parameters for the real chromosome were: population 175, 30% overlap, 1.0% mutation rate, Adewuya's method [24] for mating, and Gaussian mutation. C1 was 1, C2 was 0.1 and C3 was 0.15. There was no limit placed on the number of simulations. Listed in the following table and figure are the specifications for the antenna that resulted, as well at its gain pattern compared to the conventional Yagi. It had a score of -17.26.

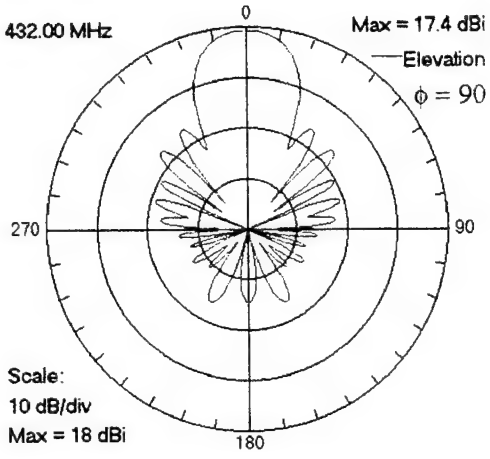
Length (λ)	Separation(λ)
0.4794	0.0000
0.4704	0.3268
0.4422	0.4866
0.4039	0.8173
0.3917	0.8790
0.4175	1.2359
0.4077	1.5694
0.3702	1.7536
0.4037	2.0358
0.3726	2.3307
0.3903	2.5751
0.3684	2.8566
0.3781	3.1037
0.3909	3.5093
0.3972	3.9214
0.3854	4.3363
0.3936	4.8036
0.4033	5.1603
Wire dia.	0.00864λ

Table 5.4. Real GA Antenna dimensions









CONVENTIONAL YAGI-18 BL = 5.16λ

GENETIC YAGI-18 BL = 5.16λ

Figure 5.8. Real GA Antenna gain pattern vs. conventional Yagi

5.16λ, 18 elements	Binary Chromosome	Real Chromosome
# of GA runs	58	15
NEC2 Sims. for the Best Run	6,124	68,412 (99% in 18,994) (99.9% in 45,439)
Best Gain	16.7 dBi	17.4 dBi
Sidelobe Level	-18.2 dB from main beam	-18.0 dB from main beam
Backlobe Level	-25.0 dB from main beam	-25.0 dB from main beam
Total NEC2 Sims for all GA runs	287,561	264,391 (5 at 7,077; 6 at about 9050; the rest with no limit)
Average score for all GA runs	-9.47	-16.4
Average Gain for all GA runs	13.9	16.4
Average SLL for all GA runs	16.0	17.7
Average BLL for all GA runs	23.3	22.9
Average VSWR for all GA runs	2.46	2.80

Table 5.5. Comparison of Real to Binary GA Optimization.

The real GA required many fewer GA runs to achieve a better result than the binary GA. Following is a table comparing the antenna performance and the simulations required to achieve them. Unlike the binary chromosome, no hillclimber was needed to finish the optimization after the GA converged.

It should be noted that the majority of the progress for the 15 GA runs included in the above table came within the first 10,000 runs. However, the runs that had over 65,000 NEC2 simulations (two of them), one at 24,000 and one at 14,000 accounted for a disproportionately high percentage of the simulations used. Also, the average score of these large users of simulations were slightly below average, at -16.2, so a huge number of simulations did not guarantee results to match.

There is a large disparity in average scores shown in the table. Since the constants are not precisely equal, scores cannot be directly compared. Thus, actual performance criteria are included in the table, and show that the real GA is better able to optimize this antenna reliably. VSWR and BLL are slightly better for the binary GA, but SLL and Gain are significantly better for the real GA. Because of this disparity, the real GA is used exclusively for the next antenna: the Arecibo Feed Yagi.

5.5 Optimization for Arecibo feed

This optimization was an unusual one for a Yagi antenna. The problem was to design a special feed for the Arecibo 305 meter spherical reflector [28]. The antenna is to be used to search for primeval hydrogen having a redshift of approximately 5. Neutral hydrogen line emission is at a frequency of 1420 MHz; thus the frequency region of interest was about 235 MHz. Preliminary studies indicated that the band from 219 to 251 MHz had relatively little interference, particularly from 223 to 243 MHz, though the interference was still quite significant. Since it was intended that the feed illuminate only about 160 meters of the reflector and since the frequency was low, it was not necessary to correct for spherical aberration. The most important design goal was for the feed to have sidelobes and backlobes at least 25 dB down in the region from $70° < \phi < 290°$, due to the interference which came from surrounding radio and TV towers. Of lesser importance was that the E- and H- plane beamwidths be about 50°. The VSWR was desired to be under 3.0 and the gain was to be as high as possible, but was limited by the wide beamwidth. The feed would be mounted over a 1.17 meter square ground plane—that is, a ground plane only 0.92 λ in size.

Since there did not seem to be any other antenna that would meet the desired specifications, it was decided to investigate the possibility of using a GA to optimize a Yagi type structure for this unconventional application.

The GA, using a real chromosome and Adewuya's method [24] of crossover and Gaussian mutation, produced a Yagi type antenna that met most of the design goals. It was quite different from a conventional Yagi in that the

director elements were very closely spaced, and it was only about a third the length of a typical Yagi having the same number of elements. This process produced the antenna that seems to best approach the above specifications, though they are quite unconventional for a Yagi antenna.

For this optimization, the GA was allowed to proceed until it was clear that it had converged. Unlike the binary GA, which eventually converges to a single chromosome and essentially stops improving, a real-valued GA will generally continue to improve the longer it is run and it rarely truly converges to a single individual. However, once the genes in the population are centered tightly around single values, this improvement becomes very slow and it becomes less and less useful to continue the GA run. Once this point was reached, the GA was stopped manually.

Following are the details specific to the Arecibo Yagi optimization. First, the number of wires was specified to be 14. The variables were: the length of each element (14 were required), each constrained to be symmetric and between 0 and 1.5λ, the spacing between each set of two elements (13 were required), constrained to be between 0.05 and 0.75λ (the total boomlength was allowed to vary in this case, unlike the high-gain optimization), and the diameter of the wire, constrained to be 2,3,4,5 or 6mm. Note that of the total 28 variables, 27 of them were continuous, real-numbered parameters, making this a natural problem for a real-valued chromosome. The discrete variable—wire diameter—used a real-valued gene, but it was discretized using truncation so the GA would only use one of the allowed values. This is usually not recommended for the type of crossover techniques used, but the problem was insensitive to this parameter and it did not affect results adversely. 175 individuals were in the population, with a 30% overlap, and 0.6% mutation rate.

The cost function used to optimize the design was:

$$f = - G_L + C_1 * SLL^2 + \Sigma(C_2 * V_i)$$

where

G_L = lowest broadside gain of all frequencies
SLL = highest sidelobe level within $70° < \phi < 290°$ for all frequencies
V = VSWR at the ith frequency. The sum is over all frequencies.

$$C_1 = \begin{cases} 0 & SLL > 25 \text{ dB} \\ 1 & SLL < 25 \text{ dB} \end{cases}$$

$$C_2 = \begin{cases} 0.1 & V < 3.0 \\ 1 & V > 3.0 \end{cases}$$

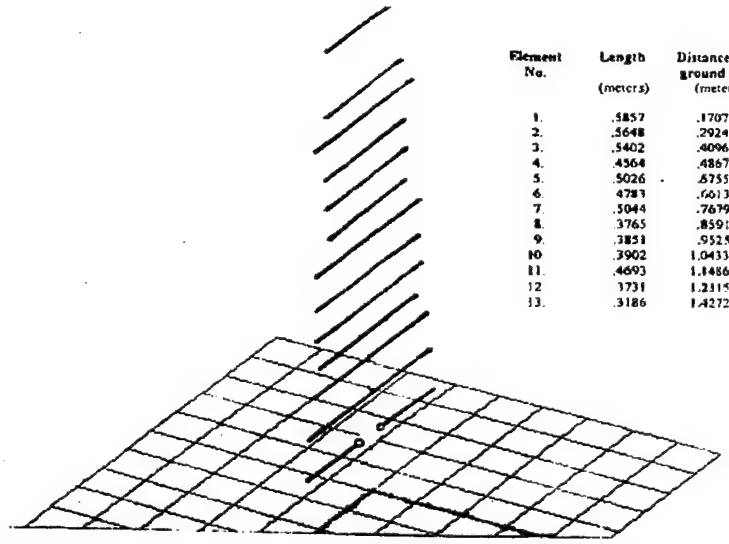
The optimization involved minimizing this function. Since beamwidth was of secondary importance, it was not included specifically in the cost function. Instead, beamwidth was implicitly defined by where the objective function began to search for sidelobes (at plus/minus 70°), and by the maximization of the gain. The antenna was simulated at 243 and 223 MHz, near the edges of the desired frequency band. It was assumed that if the antenna performed satisfactorily at these frequencies it would probably be acceptable over the rest of the band. A sample population of 175 chromosomes, with 30% overlap was chosen, and parent selection was based on the fitness-weighted roulette-wheel. The mutation rate was about 0.6%. These values were shown to be optimal by experimentation in other optimizations.

Although the feed was over a finite ground plane, it was decided to initially use a conventional reflector element in the design since modeling a finite ground plane using NEC adds a prohibitive amount of computer time. After an optimal configuration was obtained a thorough computational analysis was conducted for the whole frequency band from 219 to 251 MHz at increments of 2 MHz. This was done to ensure that the antenna was truly broadband. It was performed initially for the Yagi with only a reflector element and then repeated with a 1.17 meter ground plane.

5.5.1 Computational results

As expected, the GA produced a configuration that was quite different from one that would have been obtained using conventional methods. Typical Yagi designs have directors that are about 0.4 λ in length and 0.35 λ in spacing; the lengths become slightly shorter and the spacings become slightly larger the further the distance from the driven element. The genetic Yagi had 13 elements (plus the ground plane) with a boom length of only 1.11 λ. The directors varied in length from about 0.25 λ to 0.4 λ with an average spacing of less than 0.1 λ, as is shown in Figure 5.9. A conventional 14 element Yagi has a boom length about 3 times as long.

The performance of this Yagi was computed at 2 MHz increments over the band from 219 to 251 MHz, a bandwidth of 13.6%. The initial results were for a Yagi having a reflector element and no ground plane and are shown in Figure 5.10 for frequencies of 219, 235 and 251 MHz. Note that the sidelobes are typically higher than 20 dB down and the backlobes are even poorer, particularly at the high frequency. It seemed likely that replacing the reflector element with a 1.17 meter ground plane would reduce the backlobes and might also lower the sidelobes. Figure 5.11 shows the E- and H-plane patterns for the genetic Yagi over a finite ground plane at the same frequencies. It is seen that the side and backlobe levels for both planes are greater than 25 dB down from 223 to 243 MHz, the most important part of the band, and are over 20 dB down over the rest of the band. The E- and H-plane half-power beamwidths range from 51 to 55° and 64 to 69° respectively, slightly larger than desired but certainly acceptable. The VSWRs are less than 3.0 from 227 to 245 MHz, though they are higher at the ends of the band. The antenna gain ranged from 10.4 to 11.0 dB over the frequency band. This gain is approximately 1 dB lower than that for a Yagi that is optimized for maximum gain.



Element No.	Length (meters)	Distance from ground plane (meters)
1.	.5857	.1707
2.	.5648	.2924
3.	.5402	.4096
4.	.4564	.4867
5.	.5026	.5755
6.	.4783	.6613
7.	.5044	.7679
8.	.3765	.8591
9.	.3851	.9525
10	.3902	1.0433
11.	.4693	1.1486
12	.3731	1.2315
13.	.3186	1.4272

Figure 5.9. Genetic Yagi feed for the Arecibo Radio Telescope

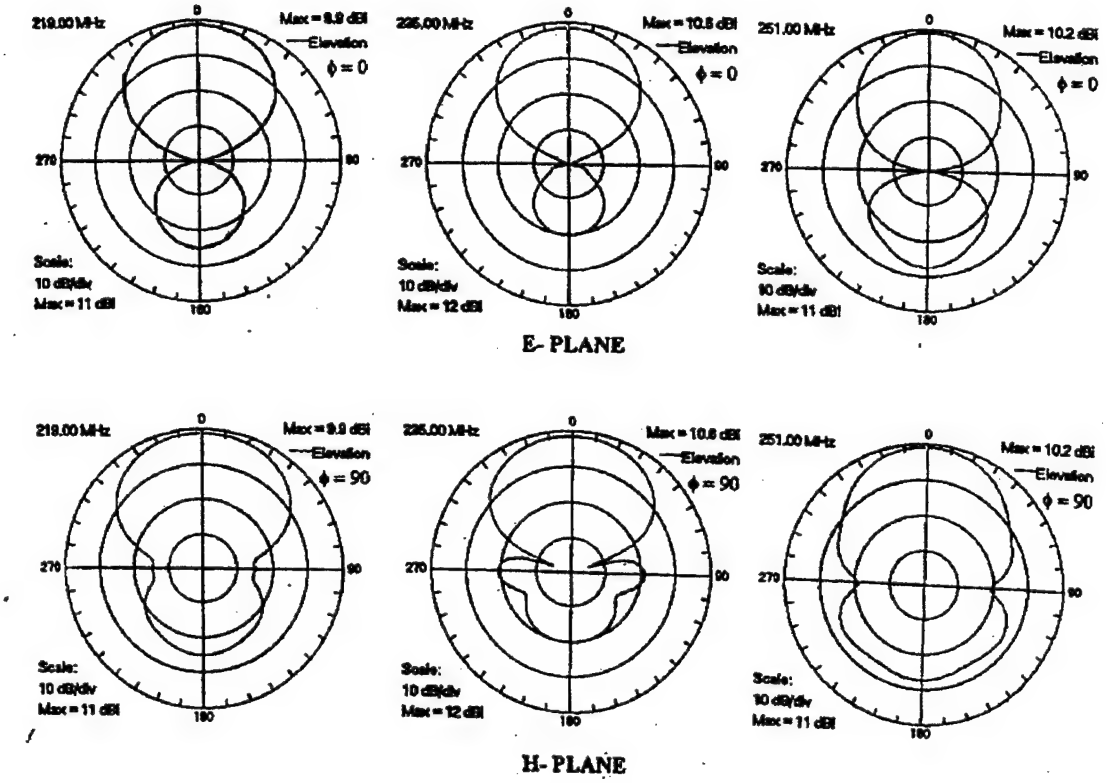


Figure 5.10. Computed antenna patterns of Yagi with reflector element at 219, 235 and 251 MHz

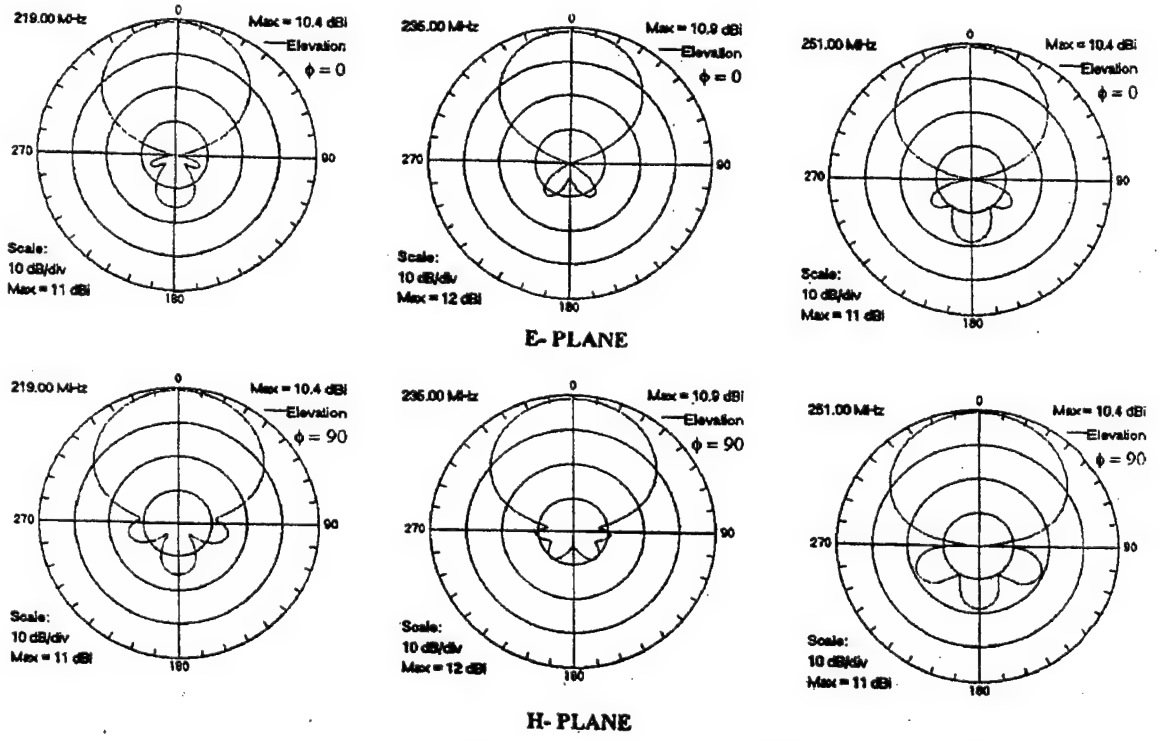


Figure 5.11. Computed antenna patterns of Yagi over a ground plane at 219, 235 and 251 MHz

5.5.2 Arecibo feed antenna validation and test

The antenna was fabricated and the E-plane patterns were measured. Since the antenna pattern range did not operate satisfactorily at frequencies below about 800 MHz and since the full scale antenna would have been quite large, it was decided to work with a 1/6th scale model having a center frequency of 1410 MHz. The Yagi elements were made of 0.8mm (1/32 in.) copper rod. These elements were inserted into 1.27 cm (1/2 in.) PVC pipe. It is estimated that the Yagi was built to an accuracy of about +/- .5 mm. The Yagi with a reflector element was first measured over the frequency range from 1310 to 1510 MHz at 10 MHz increments and then the measurements were repeated using a 19.5 cm square ground plane in place of the reflector element.

E- plane patterns and VSWR were measured for the genetic Yagi over a finite ground plane for the frequency range of 1310 to 1510 MHz at 10 MHz increments. Figure 5.12 shows the 1/6th scale model patterns for frequencies of 1310, 1410 and 1510 MHz which correspond to the full scale frequencies of 218, 235 and 252 MHz. Also shown are the previously computed patterns and it is seen that they are comparable. The measured VSWRs are less than 3.0 over most of the band and have a maximum value of 3.7 near the ends. The measured gains are slightly less than 10 dB, however, if the reflection losses are taken into account, the corrected values for a matched antenna approach the computed gains. The computational and measured gains, side and backlobe levels, beamwidths and VSWRs are summarized in Table 5.6.

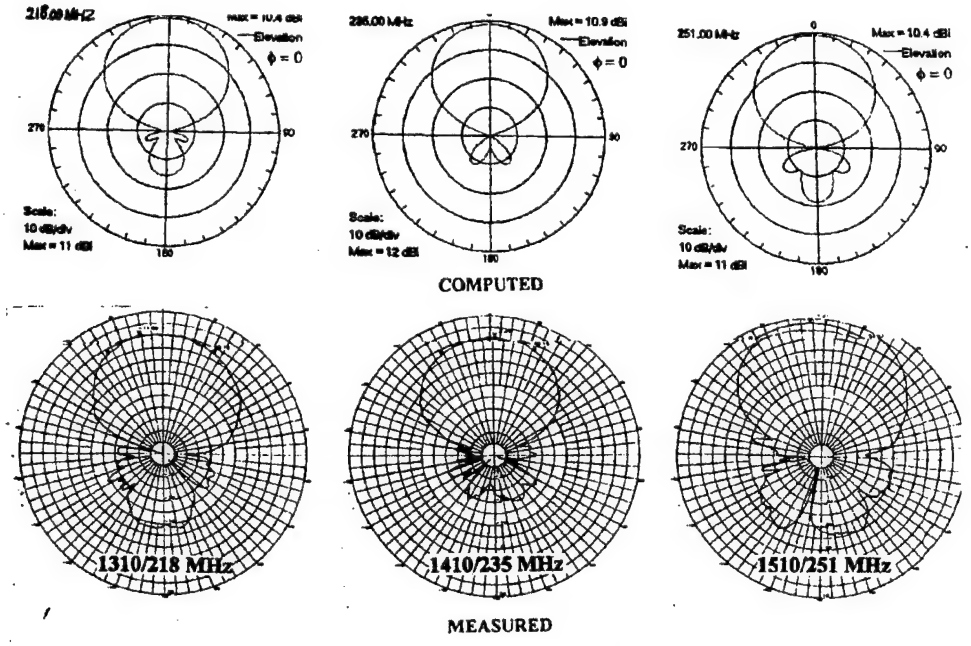


Figure 5.12. Computed and measured E- plane patterns of Yagi over a ground plane

Computed						Measured (1/6 scale model)			
Freq. (MHz)	BW E-plane (deg)	BW H-plane (deg)	SLL E-plane (dB)	SLL H-plane (dB)	VSWR	Freq. (MHz) 1/6 / Full	BW E-plane (deg)	SLL E-plane (dB)	VSWR
219	55	68	24.1	24.1	5.1	1310/219	56	18	3.6
221	54	67	24.6	24.6	4.4	1320/220	58	19	3.3
223	54	66	25.2	25.2	3.9	1330/222	58	22	3.1
225	54	66	25.9	25.9	3.3	1340/223	51	23	2.8
227	54	66	26.5	26.8	2.9	1350/225	51	25	2.4
229	53	65	26.9	26.9	2.5	1360/227	48	24	2.2
231	52	64	26.2	27.3	2.3	1370/228	53	25	2.1
233	52	64	27.4	27.9	2	1380/230	46	26	2
235	52	64	27.5	28.4	1.9	1390/232	48	25	2
237	51	63	27.5	28.3	1.9	1400/233	47	26	2
239	51	63	27.6	27.5	2	1410/235	48	28	2
241	51	63	27.7	26.6	2.1	1420/237	48	27	2.1
243	51	63	27.8	25.6	2.4	1430/238	52	24	2.3
245	51	34	27.2	24.2	2.9	1440/240	48	23	2.6
247	51	35	24.7	22.9	4.1	1450/242	42	23	2.7
249	52	66	22.6	21.6	6	1460/243	45	25	2.9
251	53	69	20.8	20.2	8.5	1470/245	48	25	3.1
						1480/247	51	24	3.4
						1490/248	50	25	3.7
						1500/250	53	22	3.7
						1510/252	53	23	3.5

Table 5.6. Summary of computed and measured parameters for the genetic Yagi over a ground plane.

5.6 Modifications of the conventional yagi antenna search space

The conventional Yagi antenna search space was modified in several ways, only one of which yielded interesting results. The elements were allowed to tilt in the plane of the antenna, but this degree of freedom did not help with gain or any other desired property. Also, elements were allowed to be asymmetric with regard to the center line of the antenna, and the drive point was allowed to be asymmetric, with similar results. If an asymmetric, unusual pattern was desired, however, these might be interesting avenues to try.

One modification did produce interesting results. As the Yagi antenna is in a single plane, it is constrained to be linearly polarized. However, if one is desiring to communicate with satellites, circular polarization is required. The elements were allowed to rotate out of the plane of the antenna to make other polarizations possible.

An optimization with a real chromosome gave a result that was circularly polarized.
The figures below show its design and its gain pattern.

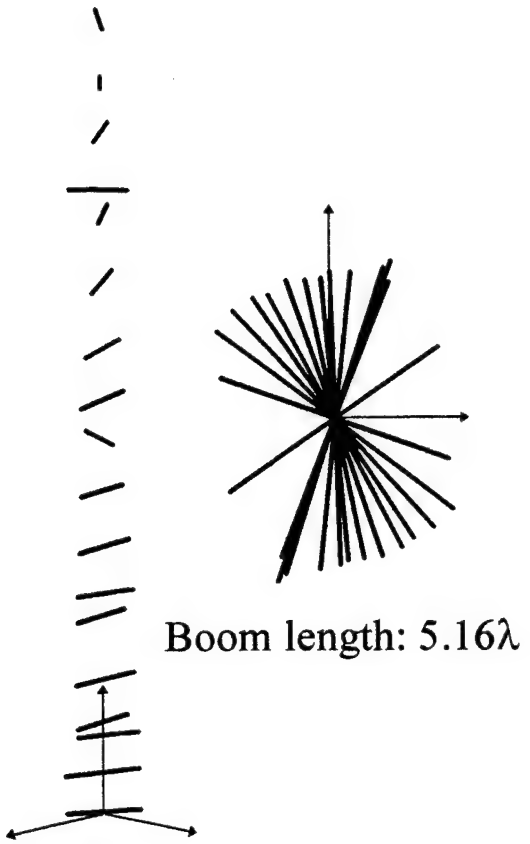


Figure 5.13. Rotated yagi—side and top views.

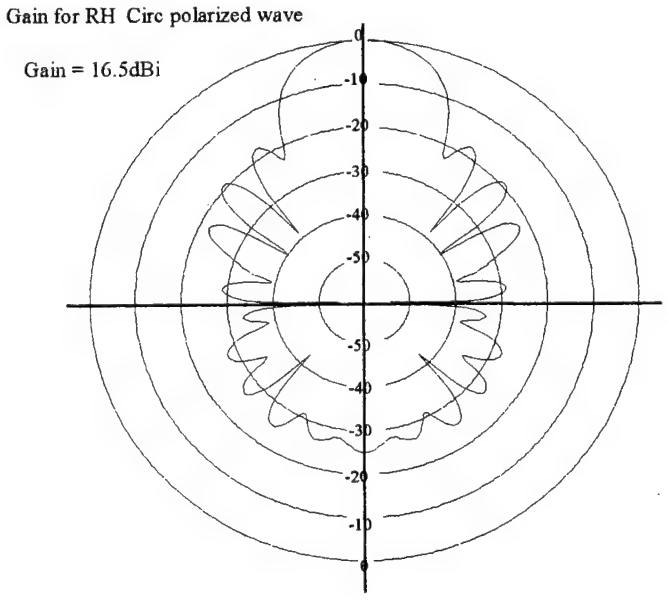


Figure 5.14. Normalized circular polarization gain pattern for rotated yagi, in the plane of the antenna (ϕ=0°). Each circle is 10dB lower than the one that encircles it. Circular polarization losses have been taken into account.

5.7 Conclusion

This chapter has discussed three different applications for the Yagi antenna: high gain, high gain with low side and backlobes, and broadband with low sidelobes. Antennas for these three cases were all atypical for Yagi antenna design, yet worked extremely well for the qualities they were optimized for. However, it should be noted that qualities that were not a part of the optimization, particularly in the high-gain case, were not as good as might be hoped. This is a lesson that is useful for many problems: if a cost function does not have all desired qualities specified, frequently those that are left out will be worse than what is wished by the engineer, though sometimes, as is the case with the next antenna, one is pleasantly surprised by these "accidental" qualities. Thus, it is important to have all desired qualities be a part of the cost function up front.

The Arecibo feed showed that a GA is capable of synthesizing a broadband Yagi antenna that has a radiation pattern that is quite different from that of a typical Yagi. This approach appears to have produced an antenna that best meets these somewhat unconventional antenna specifications. It is questionable whether an antenna having comparable specifications could have been designed using a conventional approach. Thus, the GA allows one to synthesize antenna configurations with properties that have heretofore been unattainable with existing antenna design tools.

The next chapter involves the crooked-wire genetic antenna, and the extensive research and testing of this new type of antenna. Unlike previous designs, where the basic configuration is given to the GA, the GA is allowed to configure the crooked-wire design with few constraints. This antenna shows the power of the GA to accomplish a task beyond that of mere optimization—in a way, the GA becomes the antenna engineer, forming a new antenna with a theory of operation like no other.

Chapter 6: The Crooked-wire Genetic Antenna

6.1 Introduction

This antenna is probably the most important one in this research. The reason is that it is a highly unconstrained design, very open in its shape and function, and very little information external to that discovered by the GA is available for the optimization. It is not only an interesting design, it is an excellent test of the power and limitations of GA optimization.

All other designs previously discussed have constraints that limit the possible designs to those that fall inside a particular type of antenna. In addition, the Yagi and loaded monopole antennas all have a particular method of operation in electromagnetic physics. Thus, there is a certain amount of information the GA receives *a priori*—that is, there is information about electromagnetics inside the basic antenna designs themselves that the GA does not have to discover itself. The basic antenna designs limit the applications as well, as antennas like the Yagi are basically directional, while antennas like the loaded monopole are basically broad-beamed. Of course, it has been shown with the Yagi that a design's capability can be at least in some cases be expanded beyond typical applications using a GA. Even so, its basic characteristics are still relatively unchanged.

Naturally, there is a continuum of information, from almost complete specification to a complete lack of specifications. Previous chapters have explored the former end of the spectrum. This chapter and the next will explore the latter end of this spectrum.

This antenna's interesting characteristics are primarily in its search space. To aid in exploring the characteristics of this type of optimization, where little electromagnetic information is present, its cost function has been kept simple—hemispherical coverage like the loaded monopole, but this time requiring right-hand circular polarization (RHCP) over the hemisphere. This polarization requirement makes the goal much more difficult to obtain, and more interesting.

The outcome of the GA design process working relatively unconstrained in the manner described above has been named a genetic antenna. A genetic antenna is an antenna that comes directly from a GA optimization, and is not constrained by a pre-existing design. While the GA has been used previously to help determine unknowns in existing designs, it has not been used before to create a completely new design with its own unique theory of operation. The resulting antennas described in this chapter are unique and distinct from all existing antenna designs, and were not designed using any standard techniques. As the antennas designed in this way seem to take on shapes that are different from one another but are equally unusual and non-intuitive, it seems more appropriate to give them a name that describes their origin rather than their particular shape.

After running the GA, an early resulting design was built and tested, as was done with the loaded monopole, and its radiation properties were measured, to ensure that the results, as unusual as they were, were indeed valid.

6.2 The search space

As has been implied, then, the GA does not start with a basic design like the loaded monopole for this antenna. The GA is allowed to find an antenna with the desired properties, subject to only very basic constraints, like antenna size, excitation source, and number of wires. Regarding antenna size, since near-hemispherical coverage was desired for this antenna, it should obviously be relatively small. It was decided to start by confining the antenna to a cube 0.5λ on a side. This design space is shown in Figure 6.1.

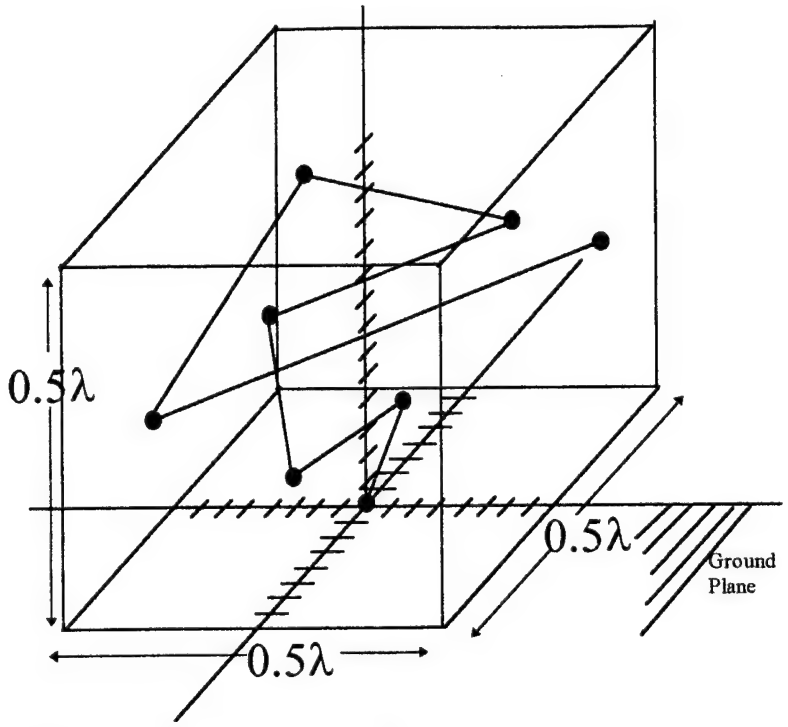


Figure 6.1. Genetic Antenna search space

Both real and binary GAs were used to design this antenna. For the binary GA, 5 bits were allowed for each component of the (X,Y,Z) coordinate for the beginning and end of each wire. In other words, each axis of the design space had 32 levels, and there were, therefore, 32^3 possible vertices at which the wires could be connected. Five bits were chosen because that allowed the GA to work in units of $1/64^{th}$ of a wavelength, which was close to the limit of fabrication tolerances at the chosen frequency of 1600 MHz. For the real GA, each parameter was made a separate real gene.

Next, the number of wires and the connection scheme that a design could use had to be specified. Initially, antennas consisting of 5, 6, 7 and 8 connected wire segments were investigated. (Preliminary results showed the 7-wire genetic antenna to perform slightly better than the 5-, 6- and 8-wire antennas, so the 7-wire genetic antenna was chosen to investigate in detail. However, both a 6- and a 7-wire genetic antenna were chosen to test initially.) In addition, it was decided to make all wires connect in series for simplicity.

With 5 bits for each axis coordinate, 3 axis coordinates per point, and 7 points to be designated (1 point per wire, since each wire starts at the previous wire's endpoint, and the first wire begins at the origin), each 7-wire genetic antenna required a binary chromosome with 5x3x7 = 105 bits. Each antenna with a real chromosome required 21 real genes. A six-wire antenna, of course, requires 90 bits. The bits for each coordinate were placed next to each other in the chromosome, as follows:

[X1, Y1, Z1, X2, Y2, Z2, X3,Y3, Z3, ... X7, Y7, Z7]

6.3 The cost function: hemispherical coverage with right-hand circular polarization

The goal was to obtain right hand circular polarization 10° above the horizon over the hemisphere at a frequency of 1600 MHz, so the cost function for this system was relatively simple. (Again, at elevations of less than 10°, the multipath components would disrupt performance, so the performance of the antenna at these elevations was not included.) The GA program used NEC2 to compute the hemispherical radiation pattern at increments of 5° in elevation (θ = -80° to +80°) and 5° in azimuth (ϕ = 0° to ϕ = 175°). The GA then reads the output of NEC2, and the

cost function calculates the average gain for a RHCP wave for elevation angles above 10°, and then calculates the sum of the squares of the deviation of all measured points from the mean. In equation form, then, the cost function is:

$$\text{Score} = \Sigma_{\text{over all } \theta,\phi}(\text{Gain}(\theta,\phi) - \text{Avg. Gain})^2 \quad (1)$$

The GA's goal was to minimize this score. A perfectly uniform gain pattern would receive a score of zero. If the average gain was less than -10 dB, the average gain was set to be -10 dB, helping the GA to eliminate antennas that were very poor radiators in general.

6.4 The binary GA

Extensive research was performed on the genetic antenna and its optimization by a binary GA. The results of this work are discussed below.

6.4.1 Initial results

In the first preliminary runs, the GA had a population of 500 chromosomes, an overlap of 50%, and a variable mutation rate. For each generation, between 0 and 20 bit-flip—0 to 1 or 1 to 0—mutations occur in the new children. Much less than 1% of the bits, and no more than about 8% of the children were affected, depending on the size of the chromosomes and the population size. The GA was halted after 90 generations.

Though the best results were obtained using the 7-wire genetic antenna in the initial runs, the 6-wire antenna was almost as good. As mentioned previously, the 5- and 8-wire antennas were somewhat poorer. The 6- and 7- wire antennas were also measured and the agreement with the computational results was excellent considering that the shape of the fabricated antenna was not identical to the model that was computed. Also the computations were done for an antenna over an infinite ground screen and the measurements were made over a finite ground screen, the effect of which will be explored. The results for the 6-wire antenna will be presented first. A 3-dimensional view of this antenna is shown in Figure 6.2. The computed results and measured results are combined onto one graph in Figure 6.3. It is seen that this antenna, like the others in this chapter, has a very weird shape in that each consists of a crooked wire going in haphazard directions. As shown by the computations, the variation is about 5dB across the hemisphere—somewhat poorer than what is desired, though much better than what most antennas can provide. The score for this antenna was approximately 450.

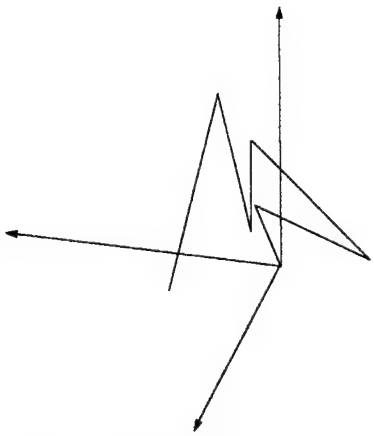


Figure 6.2. The 6-wire genetic antenna. Height of antenna is 8.66 cm.

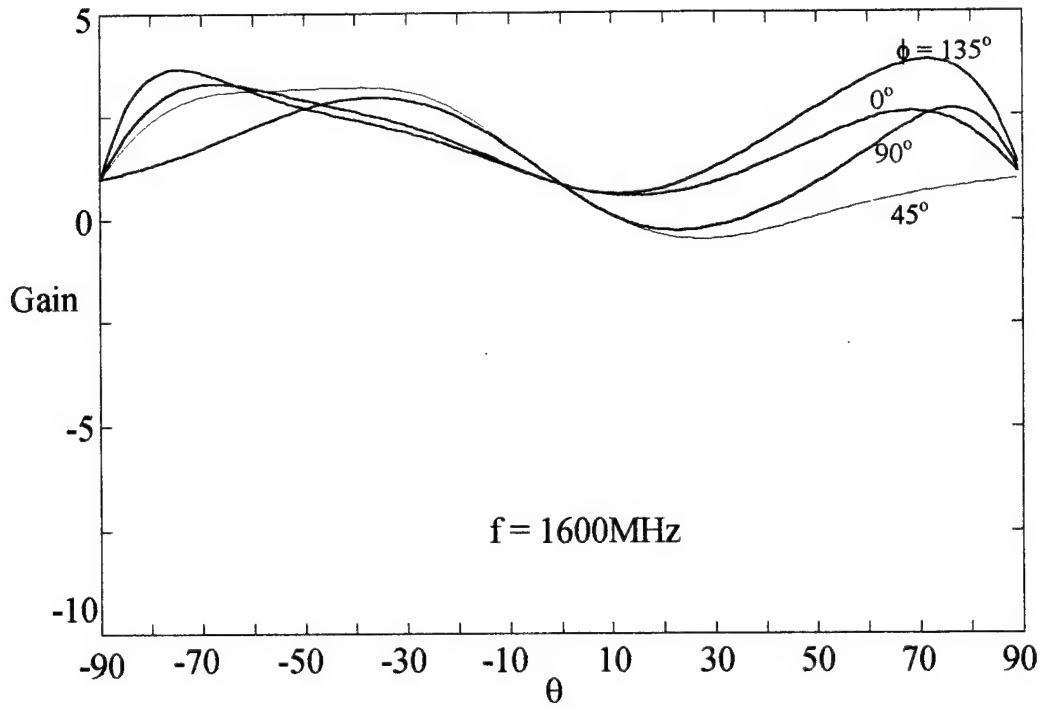


Figure 6.3. 6-element wire antenna computational results: ϕ dependence.

The radiation patterns for this first genetic antenna were measured in an anechoic chamber. The test antenna was illuminated by a right-hand circularly polarized horn antenna. θ-plane cuts were measured for azimuth angles of 0°, 45°, 90° and 135°. Because the antenna was not made exactly to specifications, it had a slightly shifted center frequency of 1550 MHz instead of the nominal 1600 MHz. The results matched very well with the computations, and are shown below.

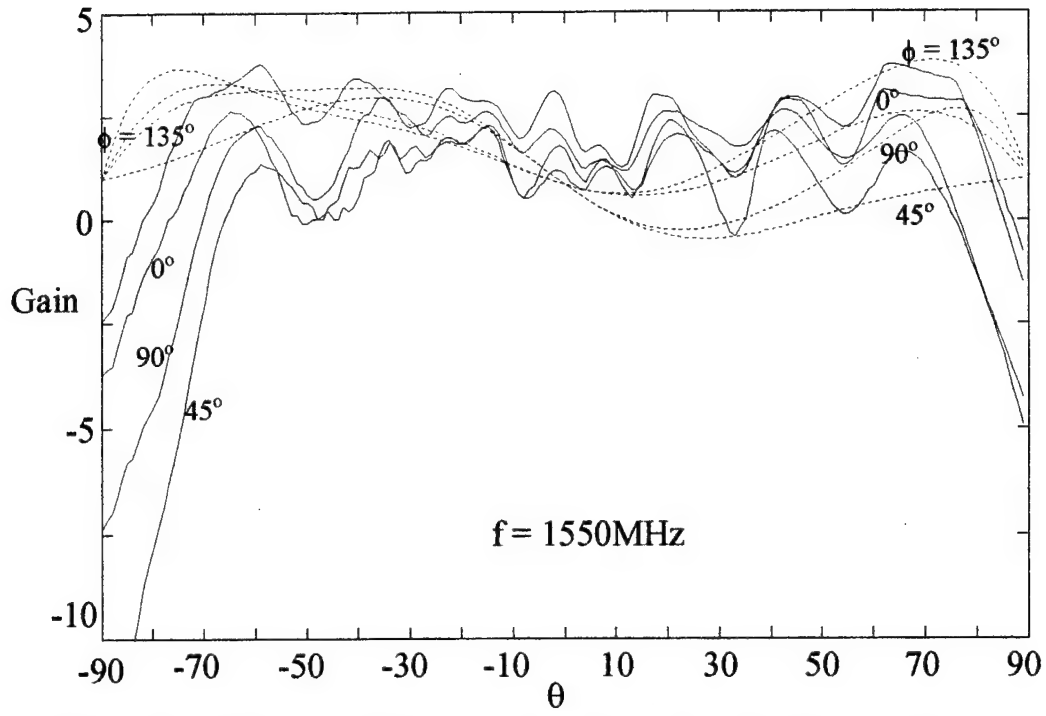


Figure 6.4. 6-element wire antenna measured results: ϕ dependence. Dotted lines: NEC output, Solid lines: Measurements.

While this antenna is good, the 7-wire design is clearly better. Like the 6-wire antenna, the 7-wire design has an unusual shape, as shown in Figure 6.5. A picture of this antenna accompanies the sketch. The coordinates for its vertices are shown in Table 6.1. The computed radiation patterns of the antenna over an infinite ground plane are shown in Figure 6.6 for elevation cuts corresponding to azimuth angles of 0°, 45°, 90° and 135° at a frequency of 1600 MHz. Note that the response to a circularly polarized wave varies by less than 4 dB for angles above 10° over the horizon. The computed frequency dependence of the radiation pattern is shown in Figure 6.7 for the range of 1200 to 1900 MHz for an elevation cut with an azimuth angle of 0°. It is seen that these patterns are relatively flat from 1300 to 1900 MHz. This corresponds to a bandwidth of over 30%, which is excellent for a circularly polarized antenna having near hemispherical coverage. Patterns for other elevation cuts were comparable to these. The score for this antenna was approximately 330, over 100 points better than the 6-wire antenna.

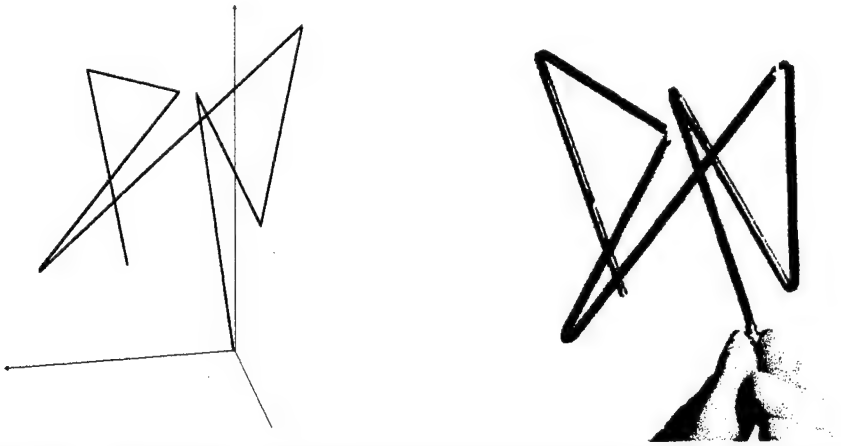


Figure 6.5. The 7-wire genetic antenna, with photograph of the actual antenna. Note that the two illustrations are at slightly different angles. Height of antenna is 8.66 cm.

7-wire genetic antenna with ground plane					
Startpoint (coordinates in meters)			Endpoint (coordinates in meters)		
X	Y	Z	X	Y	Z
0.0000	0.0000	0.0000	-0.0166	0.0045	0.0714
-0.0166	0.0045	0.0714	-0.0318	-0.0166	0.0170
-0.0318	-0.0166	0.0170	-0.0318	-0.0287	0.0775
-0.0318	-0.0287	0.0775	-0.0318	0.0439	0.0140
-0.0318	0.0439	0.0140	-0.0318	0.0045	0.0624
-0.0318	0.0045	0.0624	-0.0106	0.0378	0.0866
-0.0106	0.0378	0.0866	-0.0106	0.0257	0.0230

Table 6.1. 7-wire genetic antenna coordinates

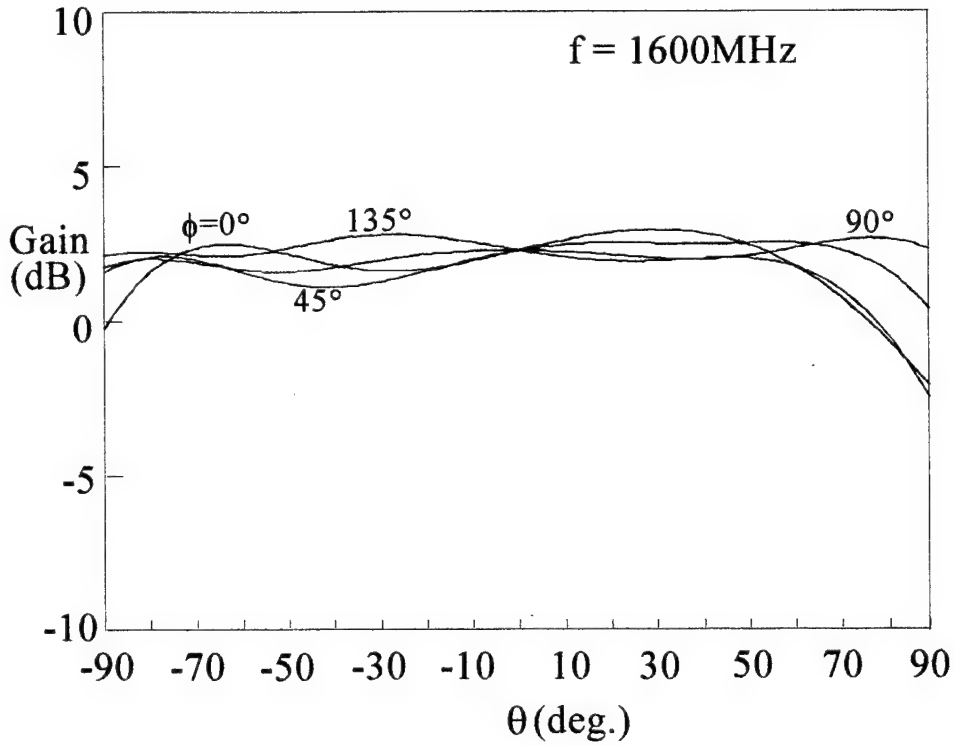


Figure 6.6. Computed ϕ dependence of 7-wire genetic antenna with ground plane.

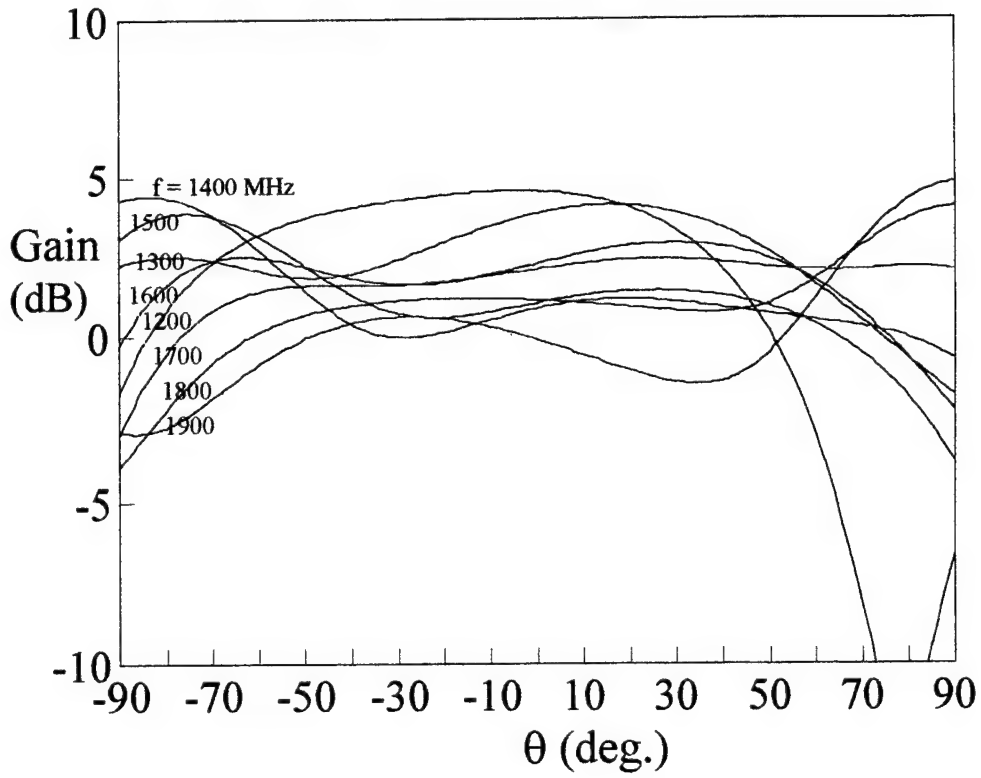


Figure 6.7. Computed frequency dependence of 7-wire genetic antenna with ground plane.

This antenna was measured in a manner identical to the 6-wire antenna. θ-plane cuts were measured for azimuth angles of 0°, 45°, 90° and 135° and for frequencies from 1200 to 2000 MHz.

The measured normalized radiation patterns for the crooked-wire genetic antenna over a ground plane are shown in Figure 6.8 for the same elevation cuts that were previously computed at a frequency of 1600 MHz. There is approximately a 6 dB variation in the field above an elevation angle of 10° as compared to the computed variation of about 4 dB. This discrepancy can for the most part be attributed to the fact that the measurements were made over a 1.2 m x 1.2 m ground plane, whereas the computations for the GA were done for an infinite ground plane. The ripples in the pattern arise from reflections from the edges of the ground plane. The roll-off at the edges is also due to the finite size of the ground.

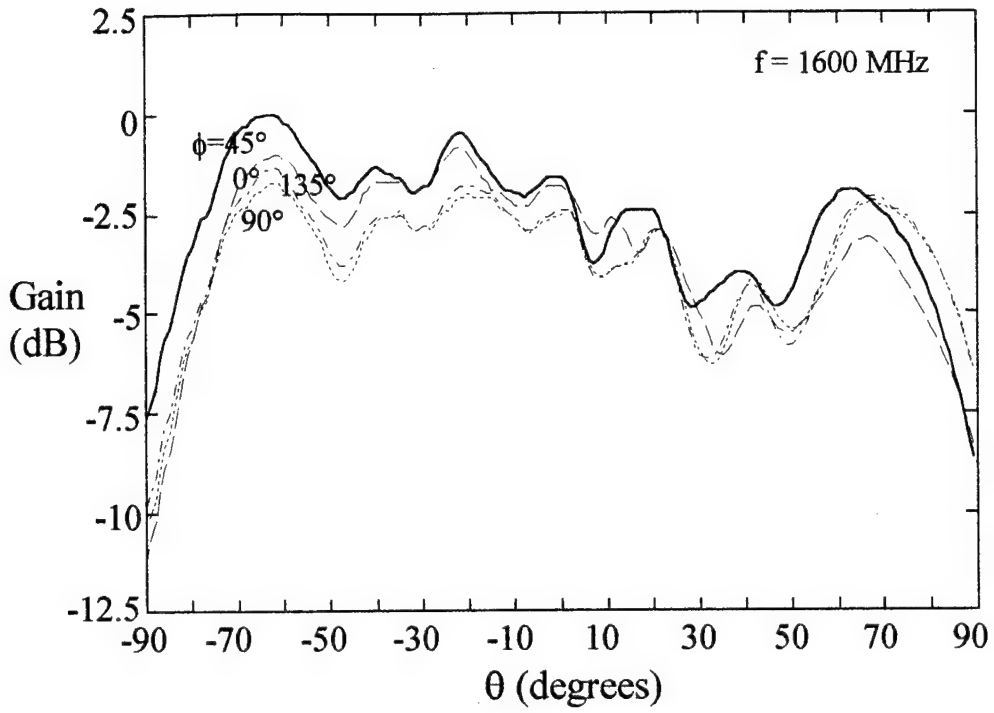


Figure 6.8. Measured ϕ dependence of 7-wire genetic antenna with ground plane.

Again, the ripples in the pattern are due to the finite ground screen as discussed in Chapter 4, section 4.?. A simulation has been performed with a finite, perfectly conducting ground screen and has produced very good agreement with measurement results, as shown in Figure 6.9. Notice that the peaks and valleys in both the measured data and the finite ground plane simulation agree well in location, though the simulation shows greater variation in amplitude. This discrepancy is minor, however, and is a result of the approximation of the real test conditions in the simulator.

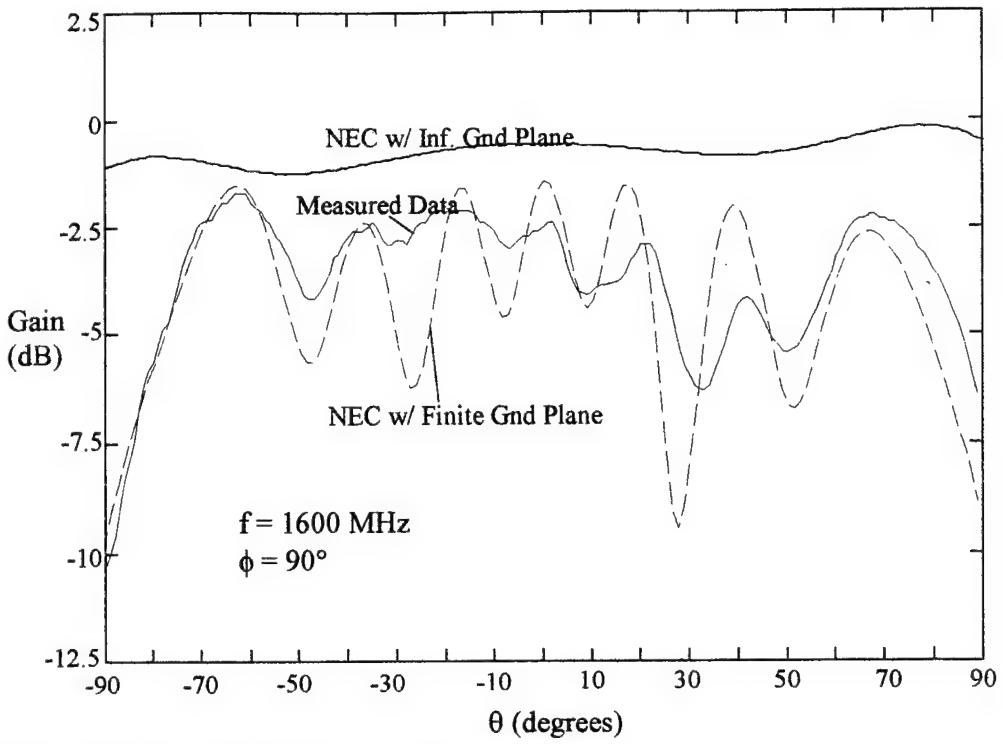


Figure 6.9. Effect of the finite ground plane on an antenna pattern.

However, NEC2 could simulate antennas over an infinite ground screen in a few seconds, while it took over 24 hours to simulate an accurate version of the 1.2m finite ground screen at 1600 MHz. Hence the infinite ground screen was used in the optimization, even though simulated performance was not identical to real measurements.

The patterns were also measured over the frequency range from 1250 to 1900 MHz for an elevation cut corresponding to an azimuth angle of 0° and are shown in Figure 6.10. It should be noted that these are relative gain patterns so only the directional properties are valid—no attempt was made to calibrate the transmitting antenna at all frequencies. As was the case for the computed patterns, these patterns show good coverage for 1300 to 1900 MHz.

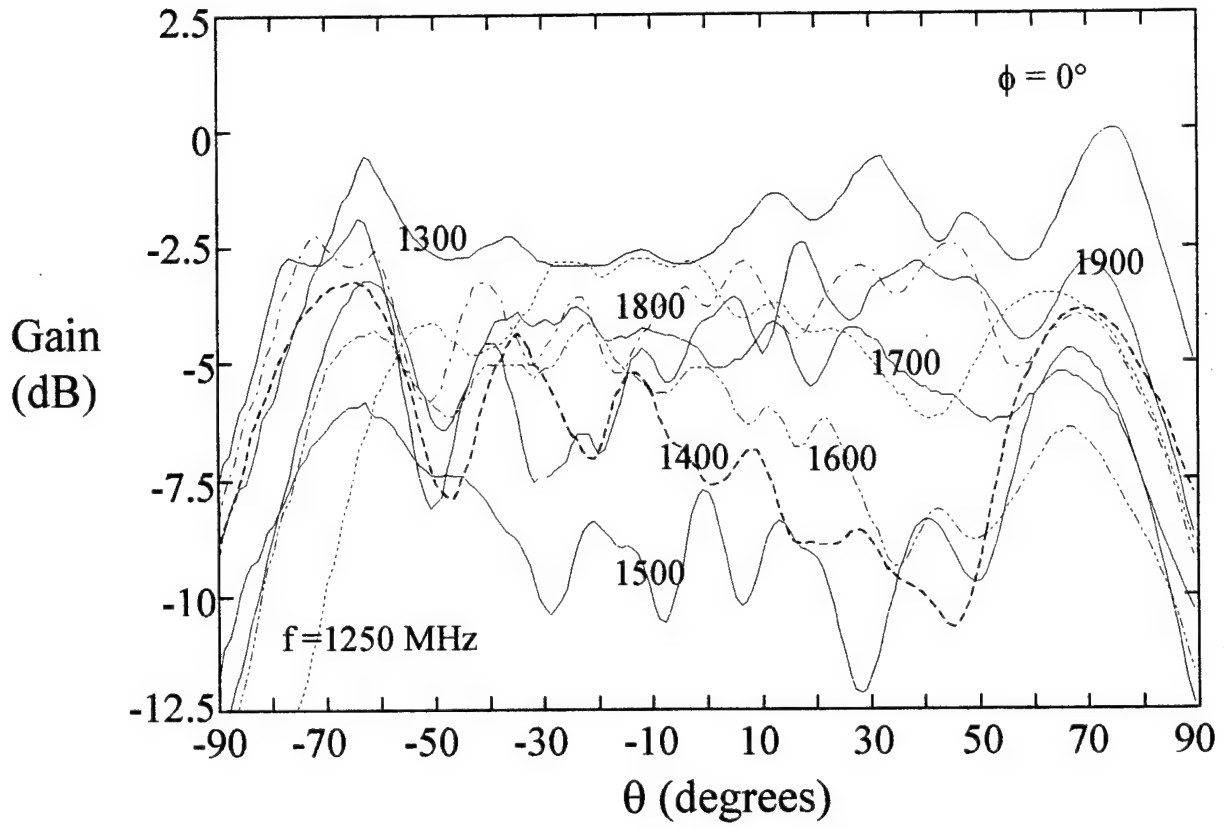


Figure 6.10. Measured frequency dependence of 7-wire genetic antenna with ground plane.

It should be mentioned that true circular polarization is not achievable over large angles. From a practical standpoint this antenna has elliptical polarization for which the magnitudes of the orthogonal signals approach unity and their respective phases approach quadrature. Note that as long as the receiving antenna has the same sense polarization as the transmitter, the maximum polarization loss of 3 dB occurs when the receiver is linearly polarized. If, however, the receiving antenna has the opposite sense polarization, the polarization loss can become very large.

6.4.3 The solution space: random results

It was of interest to determine how important the GA search method is to the discovery/design of such antennas, and how the search space behaved with respect to the variables. If the search space has a high density of good solutions, it may be possible to use a simpler technique, such as a simple stochastic search where designs are randomly generated and evaluated, or a simple stochastic hillclimber to find acceptable solutions. To explore this possibility, over 360,000 randomly generated 7-wire designs were evaluated using NEC2. The distribution of the log of the scores was found to be close to normal, with a mean of 4.337 and a standard deviation of 0.208. Thus, the average score is about 22,000. About 95% of the scores—all scores within +/- 2 standard deviations from the mean—lie between 8,300 and 57,000. The chances of randomly finding an acceptable solution is $1.6 * 10^{-15}$, implying that the expected number of runs to achieve an acceptable solution is $6.3 * 10^{14}$. An acceptable score is considered to be on the order of 800 or less in this case (meaning a 0.8 dB average variation over the hemisphere.) For comparison, the score of the 7-wire genetic antenna design found in the initial results was about 330, implying a 0.5dB variation over the hemisphere.

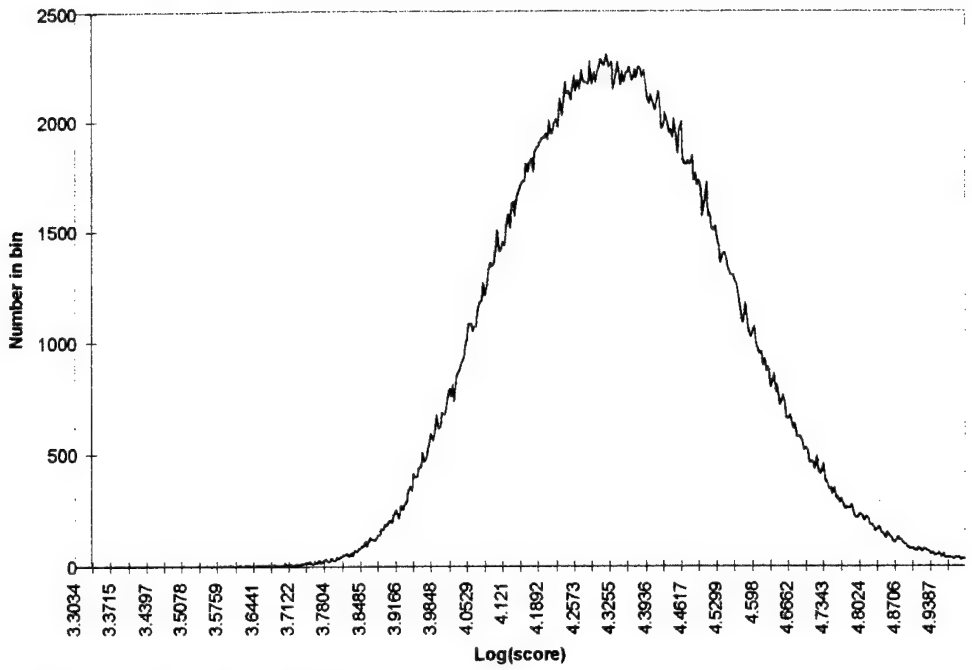


Figure 6.11. Distribution of randomly selected individuals.

It is not surprising to see this distribution, for, as with the loaded monopole, it requires special design or luck to create an antenna that is particularly bad or particularly good. However it is rather surprising to see the tightness of the distribution about 22,000, and to see that it is very close to a normal distribution with only a slight skew.

Plotting histograms for each of the individual variables as was done for the loaded monopole is not particularly enlightening. These histograms are similar to those of the loaded monopole. Though it appears that perhaps some values of some variables lower the likelihood of good results, it also seems that given any one value of any one variable, both good, bad and mediocre designs can result, and only a very small fraction of all combinations that are possible have been tried in this case, unlike the loaded monopole. To try all combinations of just two levels for each of the 21 variables would require 2^{21} NEC2 simulations (over 2 million), a prohibitively high number for even this extremely rough exhaustive search.

53 random designs out of the 360,270 were discovered to have scores less than 4000. These designs were explored with a stochastic hillclimber, requiring a few hundred runs each on average, and approximately one fourth of these explorations resulted in scores less than 800. Combining these averages implies that over 27,000 NEC2 runs would be needed to find one acceptable solution using this stochastic search plus stochastic hillclimber method. The GA technique, however, requires approximately 9,000 NEC2 runs to do the same. Though it is possible that other search techniques such as gradient methods would fare better than the stochastic hillclimber method above, the above exercise shows the relative power of the GA over a random search. It also shows the problem at hand has a very sparse solution space with many local minima, a situation tolerable for the GA, but not conducive for a classical optimization technique.

6.4.4 Experiments

There were several experiments performed with the 7-wire genetic antenna with a ground plane and the binary GA that optimized it. Because of its unconstrained nature, and the resulting lack of *a priori* information in the design, the search space for this antenna is particularly spiky and difficult to optimize. It was therefore of great interest to learn about the way a GA can best optimize this type of structure.

6.4.4.1 Experiment 1: population size, overlap, crossover

The first experiment was a coarse one, similar to that of the loaded monopole in Chapter 4. Population size was varied between two values: 50 and 500, Overlap was either 30% or 70%, and crossover was either single- or two-point. After running the full-factorial experiment with 8 replications, the following facts were discovered.

First, population size was an unquestionably important factor. A larger population led to better scores in this experiment. It also led to much higher numbers of NEC2 simulations. For the factor of 10 increase in size, from 50 to 500, scores improved from an average of 5186 to an average of 2590, and the NEC2 simulations required to converge grew from an average of 293 to 3133.

The second variable, overlap, did not make a statistically significant difference in the score of the best individual. When tested against a student's t-distribution, this variable had a t-value of only 0.46 with respect to the score, meaning it had only a 70% chance of being a significant factor given the data. To be considered significant, factors usually must have at least a 95% probability that the data supporting that conclusion is not leading to a false indication of significance. This variable could be significant, but the low t-factor indicated a strong possibility that the data could be falsely supporting that conclusion. However, if the data were to be believed to be giving a true representation of the trend, there was an indication that a larger overlap percentage led to a slightly better score, by an average of about 110 (a very minor improvement).

Though it did not make a significant difference to the score achieved, overlap did make a very significant difference in the number of NEC2 simulations required to make convergence. The effect of overlap on this response variable had a t-score of -7.1, meaning it had a significance of more than 99.95%. It indicated that a lower overlap percentage led to convergence about 900 simulations faster. Thus, a value of 30% overlap was commonly used in future GA runs.

6.4.4.2 Experiment 2: population size, mutation rate, convergence criteria, with hillclimber

A stochastic hillclimber was added to the GA to attempt to increase the exploitation of the hill found by the GA. Whether this increased efficiency, however, depended on the convergence criteria employed that would end the GA and start the hillclimber. In addition, mutation rate had not been a part of the last experiment. Thus, population size was varied among values of 100, 300, and 500, and mutation rate varied among 0.1%, 0.01%, and 0.005%. The convergence criteria variable consisted of a parameter that denoted a threshold for the amount of diversity remaining in the parent population. If the diversity dropped below this minimum amount, the GA ended and the hillclimb began. The diversity was measured by adding up the number of bits in all parents that were different from the best chromosome's bits, and then dividing by the number of bits present in the parent population to get a fraction that was constrained to be less than or equal to one. This convergence parameter varied among 0.1, 0.05 and 0.01. Overlap was held constant at 30%, and two-point crossover was used exclusively. The measured responses were: score after GA and hillclimb, and total number of NEC2 simulations required.

Once the full-factorial experiment was run with 2 replications, the results were analyzed. A quadratic model for each response was built using a statistical analysis program (RS/1 Explore), that indicated that all the varied parameters and many interactions were significant in the prediction of the number of NEC2 simulations, but that only mutation rate and population size were useful in predicting the score achieved by the optimization; score was indifferent to the convergence parameter.

The optimal settings for the GA were: population 498, mutation rate 0.1%, convergence parameter 0.0677. Running the GA 24 times with these settings gave a score averaging 1,076 with 3,436 NEC2 simulations for each, with a good answer of 800 or less appearing 11 out of 24 runs, or 46% of the time, implying an expected value of 7,496 NEC2 simulations per good antenna achieved. These results confirmed the experimental models.

6.4.4.3 Experiment 3: population size, overlap, mutation rate, NEC2 simulation limit, score limit

In this experiment, a different set of convergence criteria were used. Instead of a measure of diversity, which does not correspond well with the actual cost and goal of performing an optimization, two other parameters were used: a NEC2 simulation limit, and a score limit.

The NEC2 simulation limit can be thought of as a budget for the GA: an amount of computer resources and/or time that can be used to get the answer. As this limit is usually a major consideration in electromagnetic problems, it made sense to make this budget a direct parameter. Once the NEC2 simulation limit has been reached, the GA is allowed to finish simulating its current generation, and then the hillclimber is started. The hillclimber is allowed to get stuck as normal, since it usually only takes 10-20% of the runs used by the GA to do so.

The other parameter determining convergence is the score limit. If the GA achieves a score better than this limit, it is allowed to terminate immediately and move to the hillclimber. This allows the GA to finish significantly under budget if it is lucky in finding an acceptable individual more quickly than expected.

This experiment was more involved than those previous. It had population sizes of 800, 567, 333, and 100; overlaps of 30, 43, 57 and 70%; mutation rates of 2%, 0.43%, 0.094%, and 0.02%; NEC2 simulation limits of 400, 1600, 2800, and 4000; and score limits of 1000, 2000, 3000, and 4000. Four levels per variable allowed a 3rd order fit to be possible for each parameter. Two point crossover was still used exclusively.

A full-factorial experiment with five variables at four levels each would have been prohibitively expensive, requiring 1,024 separate GA runs, so a D-optimal design was used. This design was generated by computer, as it is a complicated, statistical way to limit the number of runs needed to formulate models as was done here. The designed experiment required only 174 runs to complete three replications.

The experiment produced two 3rd order models for the score and NEC2 simulations required. The optimal settings were population 600, overlap 40%, mutation rate 0.61%, 4000 NEC2 simulation limit, and score limit of 1000. With these settings, the GA was run 12 times, producing an average score of 909.5, using an average of 4,430 NEC2 simulations. 5 good scores were produced, giving an expectation value of 10,632 NEC2 simulations per good score. Thus, while the average score dropped, the number of NEC2 simulations increased compared to the results of Experiment 2. It should be noted that the best score yet achieved by the binary GA in any run from any experiment thus far was achieved during these 12 confirmation runs. The score was 222.75.

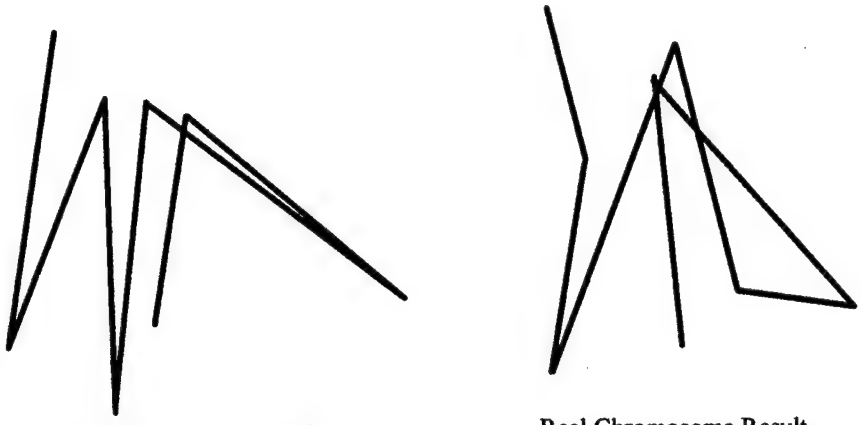
Various other experiments were tried with mutation operators, sharing functions, restricting crossover points to be on functional gene boundaries, and the like, but since none showed improvement over the standard GA, their results are not presented here. A quick study of simulated annealing as applied to this design showed that approximately 13,000 NEC2 simulations were required to achieve a decent antenna. As this was significantly poorer than the GA results, simulated annealing was not pursued further.

6.5 The real GA

As with the loaded monopole, the real chromosome was applied to this antenna after the binary GA case was well known. The crooked-wire antenna was encoded into 21 real genes, each denoting a coordinate of a point in the search space. The chromosomes were mated according to Adewuya's method. The population was 175 chromosomes, 30% overlap, mutation of 0.6%, and no limit on NEC2 runs, or score, and no hillclimb. The first run with these settings produced an antenna with a score of 185. As noted above, the best binary GA score was 222.75. This one run required 24,879 simulations to achieve this score, however, it had achieved 99.9% of the score in 24,826 simulations, and 99% at 24,667 simulations. Though this is a large number of simulations for one run, consider that it only required one GA run to beat every score produced by every binary run, which constituted hundreds of thousands of simulations.

In order to achieve these results, a modification to the normal GA mating selection process had to be made. Adewuya's method mates genes one by one. However, the unconstrained nature of the crooked-wire antenna implies high correlation between variables, higher than the other designs previously discussed. The best value for one variable, say the X coordinate for the 4th wire endpoint, is largely determined by the Y and Z coordinates for that wire and the endpoints of at least some of the other wires in the structure. Thus, it was necessary to institute a mating restriction in order to achieve good results. This restriction ensures that similar parents mate more frequently, meaning that fewer fatal children will be produced by good, but dissimilar, parents. In this way, designs are disrupted less by the gene-by-gene mating process. It was found that nothing more than this mating restriction as discussed in Chapter 3, Section 3.3.3 was necessary to ensure enough design stability in the generation of new children.

A diagram of the best binary result and the real chromosome result shows that there is only marginal similarity between the designs. This is typical of the crooked-wire antenna: its design is so open-ended that no two designs have ever been found to look alike.



Binary Chromosome Result

Real Chromosome Result

Figure 6.12. The best binary vs. best real results. Scores are nearly identical: 223 for the binary result, 185 for the real, yet they are quite dissimilar in shape.

6.6 Conclusion

This chapter has explored the crooked-wire genetic antenna with computed and measured results, search space exploration, and results of significant experimentation with GA parameters. This antenna has only been used for one cost function thus far, but it, and others like it, hold promise to solve many different kinds of antenna problems. Their unusual shape and characteristics lead one to believe they could not ever have been designed by the traditional method of antenna design. It was interesting that the antennas were quite broadband even though they were designed at a single frequency. This quality implies that the antennas are non-resonant, and that dimensions are not critical—a useful feature since the antennas were difficult to fabricate exactly.

The next chapter discusses future work with these and other types of antennas. Many possibilities exist for the engineer who wishes to explore this new type of antenna and antenna optimization, and some of the possible paths that could be followed are described.

Chapter 7: Future Work

Naturally, much work remains to be done in exploring the GA optimization of antenna designs. Though many researchers are working on optimizing existing designs using GAs, there is little, if any, work being done on unconventional designs like the crooked wire genetic antenna. In this chapter, then, are several suggestions for avenues of potentially fruitful research, focused on making the GA more efficient and effective, and on ways to explore these unconventional designs.

7.1 Future work in increasing GA efficiency and effectiveness

There are several possible avenues of future research in making the GA more effective. Techniques that have been discussed already include Adewuya's method for real chromosome mating, mating restriction, cost function construction, and optimizing GA parameters using designed experiments. However, there are several major ideas that have not yet been adequately explored: using a virtual GA for optimization of parameters, partitioning the search space in successive GA optimizations that have increasing resolution, using response-surface modeling and predictors to estimate child performance before simulation, using classical optimization methods in conjunction with a GA, and using predator/prey relationships to heighten performance. The first three of these methods have been tried in a preliminary fashion.

7.1.1 Using a Virtual GA

Greffenstette [29] has put forth an interesting idea that the GA operators like mating and mutation produce children drawn from a normal distribution, the mean and standard deviation of which are determined by the nature of the problem, but which can be predicted by the average of the parent's scores. This distribution produces children, some of which are better and some of which are worse, but all of which are drawn from the same distribution, given the same average score of the parents. Furthermore, Greffenstette found the average of the distribution for the children drops roughly proportionally to the average score of the parents.

Using this idea, a simulation can be constructed for the GA. First, the distributions for the children are found by randomly drawing parents from the search space, scoring them and mating them, and then scoring their children to build the model for the distribution. The average scores of the parents are then compared with the average of the distribution to determine the relationship between them. Information about the mutation distributions are gathered in a similar fashion, with the score of the initial individual compared with the score of the mutated one.

Once these empirical data are drawn from the search space, the first generation of a so-called Virtual GA (VGA) is drawn at random. These individuals are scored "for real" to provide a starting point for parent's scores. Then parents are mated in the usual way, and the children are created. However, instead of simulating the children, their scores are produced by a random number drawn from the distribution that corresponds with their parent's average scores. These scores are completely unassociated with the actual merits of the children's genotypes, but over the course of many children generated and many generations, they should produce an average amount of progress for the VGA. These new children then take their place in the parent population (if they were lucky enough to get high scores) and the generation process begins again, only now there are bogus results in the parent population that do not correspond to the genotype. The advantage of the VGA, of course, is that the VGA can proceed at a rate many times faster than the actual GA because, beyond the first generation, no actual scoring is taking place, only random-number generation. If the distributions for the children in the VGA are close enough to the actual distributions for the GA, then the expected optimal score of the VGA with its GA parameters of population size, etc. will be the same as the actual GA with the same, though the genotype of the "best" individual one gets at the end will have no correspondence to its actual merits.

In an attempt to find out if this approach would allow one to optimize a GA for the problems to be solved, this VGA technique was applied to the crooked-wire GA. Some limitations of Greffenstette's approach were found and some additional procedures to correct them were added to the VGA process.

First, it was found that random mating produced child distributions that had little correspondence to actual GA runs. As shown in Chapter 6, the random distribution of scores is concentrated around a mean of 22,000—a score not even close to acceptable. However, one could not simply extrapolate from the random data to get reasonable distributions as the GA progressed. As the run converged to better than random scores, the distribution of the children changed from a nearly-normal one to a bi-modal distribution, with a cluster around the low end.

In addition, it was found that the distributions were affected not only by the average scores of the parents, but by the separation between their scores. Parents with scores widely separated had more widely varying scores, while those that were closer had smaller standard deviations, in general.

Eventually, due to the complexity of these distributions, a set of distributions had to be derived from a series of 15 actual GA runs, some of which were able to achieve good scores. A look-up table was then produced for the VGA to use when determining the value of children that was able to account for the average of the parents' scores, the separation between them, and the arbitrary nature of the distributions.

In addition, the problem of clones was taken into account. As a binary GA progresses, more and more individuals are produced that are identical to their parents. The standard VGA does not account for this convergence, rather, it allows the VGA to choose new scores for the clone children. This allows for some of these clone children to have scores that are actually better than their parents', even though they are clones. This will lead to false results on average. In this implementation of the VGA, then, clones have scores that are copied from their parents.

Even with all these corrections, the VGA was not quite able to discern the best parameters, though it did come close. The VGA, with scores averaged over many runs, determined that the optimal settings were: population 400, overlap 50%, mutation rate of 0.2%, and a NEC2 simulation limit of 5000 (the only parameter that was at its limit). When these settings were used, however, these scores did not in fact produce better results than the optimal settings determined by the designed experiment in Chapter 6. Thus, it did not appear to work very well for this case.

However, this is not to say a VGA would not work well in a more well-behaved search space, and it may behoove the engineer to attempt a VGA if optimal GA settings are not easily obtained.

7.1.2 Using a series of GAs with increasing resolution

It is possible to use a succession of GA runs to optimize a single antenna, each with a limited search space that zero in on a portion of the previous GA's search space. A design is first coded into a binary chromosome using only one or two bits per variable, giving a very coarse, relatively small, search space. The first GA is run, and the best design is used to re-code a new chromosome, using the location of the best design from the first run to determine the portion of the search space that will be explored further. The result of this second run determines the search space of the third run, which is still more restrictive, and so forth, until sufficient resolution is obtained and the series of GAs is halted.

For example, if a design has two variables X and Y, with feasible limits from 0-100 for each, an initial GA chromosome could be 4 bits long, with 2 bits per variable. The levels possible with this encoding might be chosen to be 12.5, 37.5, 62.5, and 87.5, drawing each level from the center of each quarter of the range. See the figure below for a visual representation.

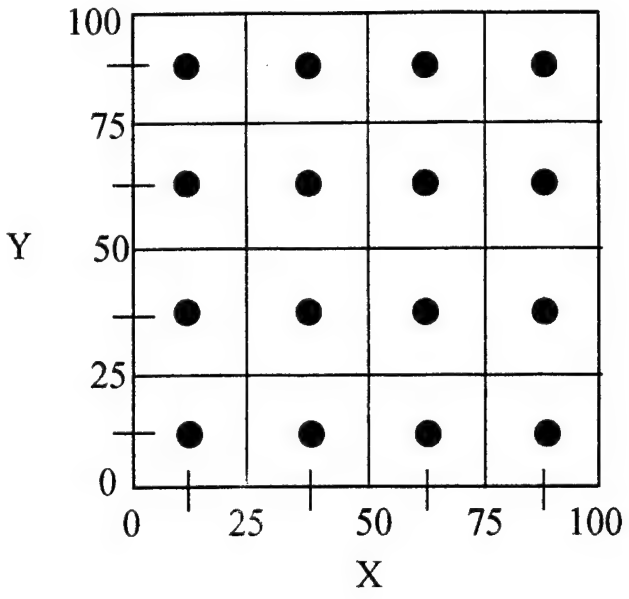


Figure 7.1. A coarse search space with levels chosen from the center of each quarter. A four-bit chromosome spans the space, and can take the values shown by the points. This representation is for the first GA run.

It should be noted that points can be taken from somewhere other than the center of each of the boxes above—they could be taken from the edges, for instance. They could even have been taken at random from anywhere in each sub-range, which would allow the GA to have a chance of finding features missed by fixed points. (Using such randomized values create a problem with efficiency and repeatability, so it was not used in actual testing of this method.)

Once the coding is in place, the first GA is then run. For the X and Y example above, let the first answer obtained be the chromosome 0110. The limits on the variables would then be changed based on this result. Since the second quarter of X was chosen, the limits on X could range from 25 to 50, giving the four levels 28.125, 34.375, 40.625, and 46.875. The chromosome could also be similarly limited to the third quarter of Y's feasible range, as shown.

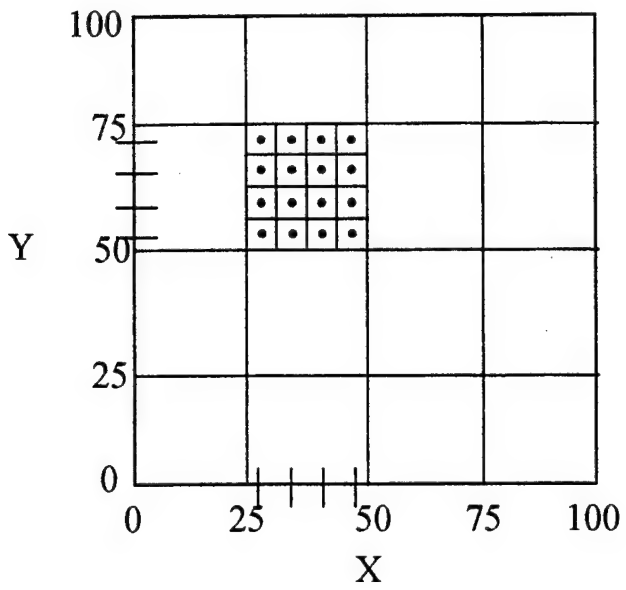


Figure 7.2. Second GA search space example. A four-bit chromosome now spans the region where Y is between 50 and 75, and X is between 25 and 50. The points indicate the possible chromosomes. This is the region containing the best result from the previous example run. Note that it is 1/16 the original area.

The number of levels could also be changed, perhaps decreasing the resolution to one bit per variable. If this were done, X levels would then be 31.25 and 43.75, and there would be four possible chromosomes. In this way, the chromosome is made still shorter, and the search space even easier to search.

Note that the number of bits per variable may be required to be more than one bit at first. There may be physical reasons to maintain a certain resolution the first one or two GA runs. However, as the search space grows smaller and more resolved, the space may need no more than two levels per variable. In electromagnetics, if the search space resolution is a significant part of a wavelength, such ¼ λ or more, there will be no way to tell if a feasible solution exists at all in the region of the search space the GA chooses—the design space will simply be too coarse for the hills to be evident. On the other hand, once the search space has come down to a resolution of hundredths of a wavelength, it is possible to have only one bit per variable, as the hills may be clearly resolved at that point, as was the case for the simple example of Chapter 3.

The range for each variable in the example above showed a drastic change from the first to second run. It need not be so dramatic. Instead of searching only the quarter in which the best chromosome was found, the best three-quarters (from 0 to 75 for X in the example) could be included in the range for the second GA. In this way, should the "true" best answer be in a neighboring quadrant to the answer found by the initial GA, one has not immediately eliminated it from consideration. This more gentle increase in resolution is shown below, for the same initial result of 0110.

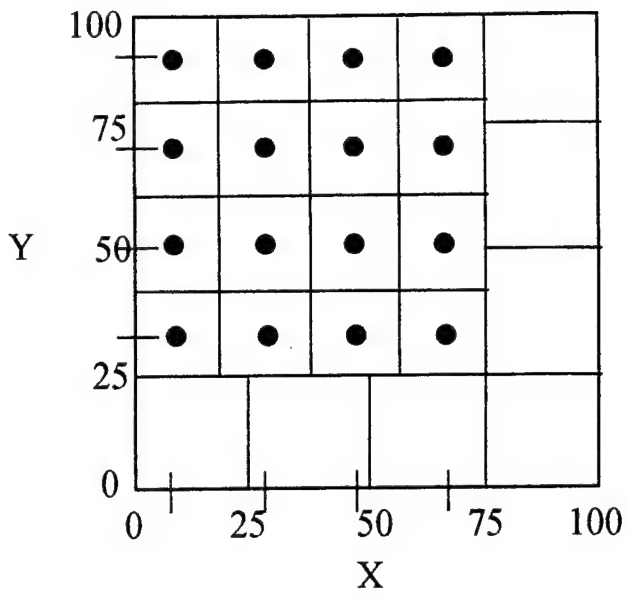


Figure 7.3. Second GA search space example, where the search space is not so dramatically limited. A four-bit chromosome now spans the region where Y is between 25 and 100, and X is between 0 and 75. The points indicate the possible chromosomes. This is the region containing the best result from the previous example run. Note that it is 3/4 the original area.

In a preliminary trial of this method, only marginal improvement was discovered. The crooked-wire genetic antenna with 7 wires was the design used. It had 21 unknowns, and was optimized for RH circular polarization and near-hemispherical coverage, as in Chapter 6. Constants were: population size 50, overlap 50%, 2-point crossover, 0.6% mutation, 250 NEC2 simulation limit. (The population size and NEC2 simulation limits were determined to be adequate in a prior experiment.) Each optimization used a series of 5 GA runs. The unknowns for each successive run was limited to the quarter or half in which the previous GA's best individual was located, similar to what was shown in Figure 7.2. A stochastic hillclimb was performed at the end of each GA run, generally improving the score between 0-30%, often with less than 100 simulations. The trial was to determine how many of the series of 5 GA runs required a resolution of two bits per variable in order to have the best performance. The remainder of the GA runs used one bit per variable.

Number of GA runs with two bits per variable	Avg score	St Dev	Lowest score	number of good answers (scores < 800)	number of total optimizations	fraction that have good answers (scores < 800)
0	2223	1262	1244	0	6	0.000
1	3148	1663	534	3	16	0.188
2	2944	1393	404	2	16	0.125
3	2741	1568	354	2	10	0.200
4	2789	1110	1355	0	6	0.000

Number of GA runs with two bits per variable	Avg number of simulations	St Dev	Smallest number of NEC2 simulations
0	1136	111	1025
1	1239	92	1109
2	1327	95	1168
3	1519	145	1242
4	1620	56	1528

Table 7.1. Results with series of GAs with increasing resolution

The average number of NEC2 runs necessary for each 5-GA series was 1361, which is much smaller than the average number of runs for a normal optimization—about 4,500. However, using only those cases (1,2, and 3 GAs with two bits per variable) that had good answers, the fraction of runs that had good answers was 17.1%. This implies that 5.9 full optimizations were required per good answer, which gives an expected value of 7970 NEC2 simulations per good point, compared with about 7500-10,000 for the normal binary GA method, depending on GA parameters. Thus, no improvement was achieved.

Further experimentation revealed that having three GA runs in the series of 5 with two bits per variable, with the other two having only one bit per variable, was actually the optimal combination. Using this combination alone improved the average number of NEC2 runs per good point to about 6,200, which is an improvement of 17% over the minimum 7500 simulations for the normal binary GA.

One way to significantly improve this method's performance is to monitor the first GAs results and start it over if it converges to a score that is too high. Unlike the normal GA, it is easy to see whether it is of benefit to continue a particular series, or if it is best to halt the series and start over from the full search space. A result above a threshold score in the beginning or middle of the series indicates that the GA is limited to a poor part of the entire search space, and that the series should be re-run. This saves one from having to run the series to completion, using many simulations, only to find out the final result is not acceptable.

The threshold score for the first GA of the series seemed to be 3000 for the 7-wire genetic antenna. If the first GA could converge to a score less than 3000, then later runs were likely to find good hills to exploit, and the whole series was likely to converge to a good result. Eight series were run using this threshold score (if the score was above the threshold, the first GA was run again) and, out of the eight series that used a total of 20,650 NEC2 simulations, six good antennas resulted. This gives an average of only 3441 runs per good antenna—an improvement of 54% over the normal GA!

Once this threshold score was discovered and verified, it was desired to decrease the number of repeat first GA runs that would be required to discover the first score less than 3000. With the parameters listed above (pop 50, overlap 50%, mutation 0.6%, 250 NEC2 simulation limit, hillclimb used), it required an average of 6 GA runs and a total of 1,754 simulations to find the first good individual from which to restrict the search space.

An experiment was then conducted, which varied the population size between 50 (with a 250 NEC2 simulation limit) and 100 (with a 500 NEC2 simulation limit), and the number of bits per variable between 2 and 3. Using about 500 separate GA runs, it was determined that a population of 50 with 2 bits/variable was optimal, and required an average of 1515 NEC2 simulations to find an individual with a score less than 3000. The next best parameter settings were a population of 100 with the 3 bits/variable, requiring an average of 1728 NEC2 simulations per individual with a score of less than 3000.

This method of gradually restricting the search space, then, shows promise as a way to limit the number of simulations needed in an optimization. It can be likened to a coarse digitization of a television picture—at first one can only see large features, like the outline of a person or a building, which may be enough to indicate where one should increase the resolution to find the desired feature like an eye or a window. The search space is able to become much more manageable with this method, making the 7-wire design chromosome only 21 to 42 bits long, with 1 or 2 bits per variable, respectively, as opposed to the original 105 bits.

How well this method works on other problems is unknown. Some problems will require more resolution than others to resolve the major features of the search space. In addition, it is not known why it was best to have only two bits per variable for the first three runs, instead of the first two or even just one. At first glance, it would seem the most likely GA run of the series to need high resolution is the first one. After this first one, the search space is much, much smaller—with two bits, the 7-wire space became $1/(4.4 \cdot 10^{12})$ of its former size, and had a resolution of 1/16 λ. Why it would still require a resolution of 2 bits per variable (giving a resolution of 1/256 λ) in the next level is not understood, let alone the 1/4096 λ resolution of the third level. Thus, this is an area that would benefit from further research. The reader is referred to [32] for another work that investigates this concept.

7.1.3 Using response-surface modeling for child pre-processing

Another possible way to increase GA performance is to allow children's performance to be predicted before they are simulated, thereby eliminating likely poor performers before a costly simulation is wasted on them. To keep the problem simple, the loaded monopole served as the example on which to test this method.

To implement this method, the GA process proceeds as normal, randomly generating and simulating the first generation, until just prior to producing the first children. At this point, a number of chromosomes are used to create a multilinear regression. This regression is based on the gene values and the scores of the parents, and creates a model for the search space. The children are then produced in the normal way, but instead of simulating them, the regression model is used to predict their performance, based on their gene values. The children and parents are then rank-ordered using the predicted scores of the children and the real scores of the parents. If children end up in the parent population, it has been found crucial to simulate them in order to find out their actual scores. If they do not perform as predicted, the population is rank-ordered again, and any other children that have moved into the parent population are simulated. This rank-ordering and simulating process repeats until there are no more unsimulated chromosomes in the parent population. Then children are generated and the process is begun anew. In this way, it is hoped that poor performers will never be simulated, which will cut down on the number of simulations needed to achieve good results.

A designed experiment was performed to find the optimal parameters for this process. The variables were: population size (75, 150 and 300), overlap (30%, 70%), mutation rate (2%, 0.2%), number of chromosomes used in regression (the most recent 75, 150, or 250 chromosomes), basis function type (sines and cosines, polynomial), and order (2^{nd}, 4^{th}). After performing the full-factorial experiment, the optimal settings were found to be: population 210, overlap 30%, mutation rate 0.63%, the 75 most-recent chromosomes used for the regression, and 2^{nd} order polynomial basis functions. Scores obtained with these settings averaged 40.8, with 3,034 NEC2 simulations to acquire them. As discussed in Chapter 4, scores such as these required about 4,040 NEC2 simulations to achieve using a normal binary GA, so there is about a 25% savings in runs.

Ways to improve this method include limiting or eliminating extrapolation. It was found that often those that were predicted to have phenomenal scores failed miserably. This was due to extrapolation: the data that the regression was built upon often did not include a large enough span of the search space to include all children that resulted. Hence, some were predicted outside the span of the regression, and were frequently predicted to be much better than they were. If this could be corrected, either by disregarding scores that appear to be too far above the norm, or by measuring the distance of a child from the regressed region of the search space, one might be able to limit the waste of simulating poor children.

Additionally, if one can confidently predict children's scores with accuracy, one may be able to never simulate some of them at all, even if they are predicted to be good. In order to do this, the validity of the model must be known, as well as the likelihood that the child falls within the limits of the regression and is not residing on a feature of the search space missed by the regression. This method shows promise in providing a way to drastically reduce the number of simulations required to find a good design, but much work remains to be done to realize these savings.

7.1.4 Using classical optimization methods with the GA

Another method of improving the efficiency of the GA is to couple it with a classical optimization strategy. As was discovered with the crooked-wire and Yagi antennas, the GA is much more capable at hill finding than hill climbing once it has converged to a large extent. A stochastic hillclimber was used to finish the hill climbing, but as it is a random technique, it is inherently inefficient. A classical method like a simplex or conjugate gradient hillclimber is probably a much better option for this final stage in the optimization, since the GA has provided the equivalent of a very good guess at a final solution, and the problem is assumed to be unimodal at that point.

Unfortunately, it is not possible to take any sort of gradient analytically with these types of structures. It will be necessary, then, to approximate such gradients, but doing so will decrease the efficiency of classical methods that require them.

7.1.5 Using predator/prey

In biological life, species must contend, not only with the physical environment, but also with other species. Particularly powerful in the evolution process is the "biological arms race" between predators and their prey. Predators attack their prey and generally kill the least-fit individuals, leaving those with better defenses to survive and mate. The improved prey can now avoid the attack of the predators more effectively, making the least-fit of the predators unable to kill enough prey for food and eliminating them from the predators' gene pool. Consequently, the best predators mate and produce offspring that are better adapted to counter the prey's defenses. Thus, both species evolve more rapidly than those that must adapt for their environment alone. This is so because the environment generally changes very slowly, and once a species has adapted sufficiently to overcome its obstacles, its progress will stagnate.

This same approach can be taken with GAs. Predator GAs can co-evolve with prey-type GAs. Scores for the prey are generally taken from their effectiveness at solving a problem, predator scores are generally taken from their ability to exploit weaknesses in the prey's solutions, making their prey's scores lower.

How this situation can be applied to an antenna situation is tricky, however. One does not want to put the constants of a cost function completely into the hands of the predator GAs, for example, for the predator will soon discover that the best way to destroy an antenna's score is to put each constant of the cost function at its maximum value. Perhaps a better way is to limit the total of all constants the predator can use in the cost function. In doing so, the predator must find the worst characteristic of the prey and put a larger constant on that term in the cost function. The prey will improve this characteristic, and the predator will have to readjust the constants. This is but one possibility; there are countless others.

7.2 Future work in unconventional genetic antennas

There are limitless ways to configure antennas. If one allows for optimizations without pre-existing designs like the crooked-wire, one can use almost any configuration of wires connected in series and parallel, perhaps with some not even connected at all. However, it is possible to have the avenues to explore delineated by a nomenclature that suggests them to the designer. Hence, a possible taxonomy for genetic antennas without existing designs has been constructed. By naming and classifying these complicated structures, it is hoped that future researchers will be able to explore the various possible structures with some measure of thoroughness and organization. The surface of the vast realm of genetic antennas has just barely been scratched, having only one major type of genetic antenna search space explored thus far, as they are delineated in the table below. After the table and a brief description, a few different ideas are discussed: multi-chromosomal antennas, tree antennas, and using developmental rules. The first two of these ideas have been explored briefly.

7.2.1 Taxonomy for genetic wire antennas

Name Category	Name	Search Space (SS) defn.	Resulting Antenna (RA) Defn.
Number of Points	point	a point in space that is used as a location for one or more joints or a wire termination.	
	n-point	maximum number of points that can be used	number of points used in the RA as locations for joints
	strictly n-point	SS constrained to include n points for every antenna	n/a
	omni-point	no constraint on the number of points used in SS	n/a
Number of Wires	n-wire	maximum number of wires possible for RA	number of wires between points present in RA
	strictly n-wire	SS constrained to include only antennas with n wires	n/a
	omni-wire	no constraint on number of wires	n/a
Wire	straight-wire	wires in SS are all straight	wires in RA are straight
	nth order wire	wires are curved using at most nth order polynomial equation	wires are curved using at most nth order polynomial equation
	strictly nth order wire	only nth order polynomials are used to determine wires	only nth order polynomials are used to determine wires
	nth order spline wire	wires are curved using at most nth order spline	wires are curved using at most nth order spline
	thin/thick-wire	wires are thick or thin compared to their length (thin is chosen if NEC is used)	wires are thick or thin compared to their length (thin is chosen if NEC

			is used)
Jointedness	joint	a connection between a designated wire and another wire.	
	n-jointed	up to n joints can occur at a single point, which is to say that up to n+1 wires can be connected at a single point. If three wires are joined at a point, it is double jointed. If five wires are connected, it is quadruple-jointed.	n is the maximum number of joints that occurs at any one point in the RA
	strictly n-jointed	n-joints are constrained to occur at all points except wire terminations	all joints connect only n+1 wires. An antenna composed of a single series of wires would be strictly single-jointed.
	0-joint	occurs when a wire (or group of wires) cannot trace a connected path to a signal source. These wires are parasitics. Strictly speaking, 0-joints occur at wire terminations (one wire goes into a point, and none go out) but are not counted as 0-joints as wire terminations are not special features.	
	non-0 jointed	0 joints (i.e. parasitics) are not allowed in the SS	RA has no parasitics
	multi-jointed	points can contain more than one size of joint	points in the RA do not all have the same size of joint (excluding wire terminations)
	omni-jointed	no constraints are placed on jointedness at points	n/a
	non-terminated (i.e. non-0 jointed to include wire terminations)	no wire terminations allowed, only connected-wire loops	no wire terminations exist in RA, only connected-wire loops
RLC Loading	Unloaded	No RLC loads used	No RLC loads used
	n-(RLC)-loaded	up to n loads are available, with R,L and/or C impedances	n loads are used in RA
	strictly n-(RLC)-loaded	n loads used in each antenna	n/a
	omni-(RLC)-loaded	no restrictions placed on number of RL and/or C loads	n/a
	in series	R,L,C elements in a load are in series with each other	R,L,C elements in a load are in series with each other
	in parallel	R,L,C elements in a load are in parallel with each other	R,L,C elements in a load are in parallel with each other
	at joints	loads are placed at joints	loads are placed at joints
	on wires	loads are placed on wires	loads are placed on wires
Dimensions	n-dimensional	n dimensions exist in the search space (1, 2 or 3)	the RA has 1, 2 or 3 dimensions
Geometry	cubic/spherical/etc.	SS is located inside a cube/sphere/etc. (3-D)	n/a
	square/circle/etc.	SS is located inside a square/circle/etc. (2-D)	n/a
	arbitrary	Geometry is unspecified	n/a
Size	n-λ	SS is limited by n wavelengths	Size of RA in λ
	n-meter	SS is limited by n meters	Size of RA in meters
Resolution	n-λ	chromosome can resolve point coordinates to n wavelengths	n/a
	n-meter	chromosome can resolve point coordinates to n meters	n/a
Ground	perfect/ imperfect ground	SS is over a perfect/imperfect ground	RA is over a perfect/imperfect ground
	finite/infinite ground	SS is over a finite/infinite ground	RA is over a finite/infinite ground
	no ground	no ground is used in SS	no ground used for RA
	unspecified ground	ground to be determined by GA	n/a
Chromosome	n-chromosome	Individuals have n chromosomes	n/a
	binary string	Chrom. is binary string	n/a
	real number string	Chrom. Is string of real numbers	n/a
	tree (___ in nodes)	Chrom is tree (with ____ objects (such as point coordinates) in its nodes)	n/a
Optimized for...	bandwidth	cost function includes terms measuring bandwidth	
	nulls	same as above for null pattern	
	sidelobes	same as above for sidelobes	
	beamwidth	same as above for beamwidth	
	impedance match	same as above for impedance matching	

Table 7.2. Taxonomy for genetic wire antennas.

The crooked wire antenna, then, would be classified as having a strictly 7-point, strictly single-jointed, straight/thin-wire, single-chromosome, cubic 0.5λ search space with a perfect ground plane, optimized for beamwidth. In the binary case, the classification would also include a 0.0156λ resolution binary string (from which one could deduce the number of bits used per variable), while in the real case, the classification would include a real-number string. The resulting antenna would be a 7-point, 7-straight-wire, strictly single-jointed, 3-D genetic antenna over an infinite, perfect ground plane.

As another example for using this taxonomy, consider an antenna produced with a tree chromosome. For instance, one such antenna search space could be classified as 10-point, straight/thin-wire, 4-jointed (implying multi-jointed, but not omni-jointed), 3-dimensional, arbitrary, and having a single tree chromosome (3-D points in nodes) and an infinite, perfect ground plane. An example antenna resulting from this search space could be a 10-point, double-jointed, straight-wire, 3-dimensional, 10cm genetic antenna over a perfect, infinite ground plane.

Though there are a number of terms that describe a single antenna, using all these terms will allow the researcher to clearly state what type of search space the antenna was designed from, and hopefully will allow him or her to make correlations between configurations and performance. It provides a way to frame the large number of qualitative and quantitative variables present in these optimizations, so that if a particular search space is not working, another can be chosen in a logical fashion. It is expected that the taxonomy, as extensive as it is, is most likely still inadequate to completely describe the realm of genetic antennas and will require expansion, as very little has been done to explore the effect of different search space sizes, shapes, and connection schemes. The rest of this section is therefore devoted to describing ways to explore these search spaces that seem promising.

7.2.2 Multi-chromosomal antennas

One major exploration would be into multi-chromosomal GAs. In biological life, all creatures have more than one chromosome in their genetic makeup. It would be of use, then, to explore adding multiple chromosomes, as is suggested in the taxonomy above. In so doing, genes would be allowed to independently assort (though this already happens with real chromosomes that crossover gene-by-gene). It would also be possible to mix real and binary chromosomes together in a single antenna.

This multi-chromosomal configuration has been explored briefly. A multi-chromosomal individual was set up in the normal GA. Each chromosome has a set of information: one denotes a set of 7 points in 3-D space, generated in the same manner as with the crooked-wire. The second chromosome contains connection information for eight wires. This information takes the form of 6 bits for each wire—three bits to denote which of the eight points in space (7 points in the chromosome plus the origin) will be the starting point for the wire, and three to denote the end point. The only exception is the first wire, whose starting point is constrained to be the origin (since at least one wire must be connected to the signal feed.) Also, wires that duplicate other wires are removed from the input file. This setup allows the antenna phenotype to take a large number of configurations, with wires in series and parallel. Points can have all the wires attached to them, or no wires connected at all. Wires can be connected to the active structure, or float in space as parasitic elements.

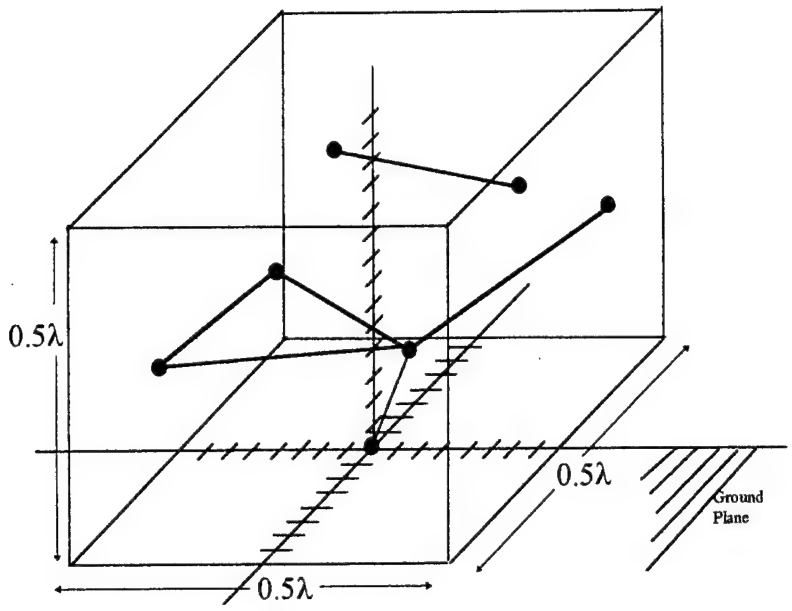


Figure 7.4. Multi-chromosomal 3-D Design Space.

As with the crooked-wire antenna, a 3-D cube of space is available with 32 levels possible in each of three dimensions for a designated number of points. Wires are connected between any two points, with the following constraints: 1. the first wire must start at the origin, and 2. a wire is not drawn if it duplicates another wire or if the two endpoints are the same. In this way, wires can be somewhat easily eliminated from the design without leaving trash in the chromosome. Again, the ground plane can be removed if desired, and the limits on size changed. When applied to the RH Circular Polarization, hemispherical coverage problem, results were similar to the crooked-wire antenna, though the shapes were naturally quite different.

A 2-D genetic antenna version of the above space has been investigated minimally.

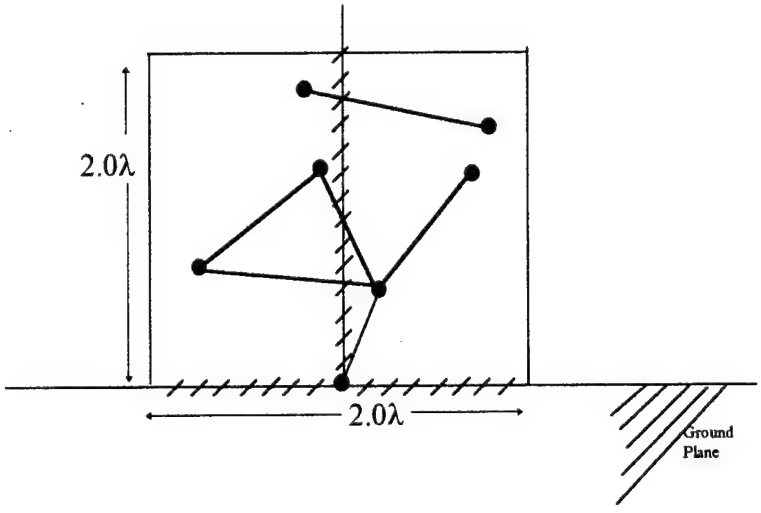


Figure 7.5. 2-D Multi-chromosomal Design Space.

A 2-D plane with 32 levels possible in each of two dimensions for a designated number of points was searched. Again, wires are connected between any two points, with the following constraints: 1. the first wire must start at the

origin, 2. a wire is not drawn if it duplicates another wire or if the two endpoints are the same, and 3. a wire is not drawn if it intersects any of the previously drawn wires. NEC2 does not know how to handle wires that intersect at points other than the ends of segments, so wires intersecting at arbitrary locations can cause simulations showing no gain at all, or falsely high gains (though the directivity shown is often still correct). The number of completely infeasible solutions NEC2 simulated was limited using rule 3. Again, the ground plane can be removed if desired, and the limits on size changed.

The results of optimizing this structure for the Arecibo Feed problem as posited in Chapter 5 were somewhat disappointing, however. It appears that the optimization of this structure is rather difficult, though it is acknowledged that the Arecibo Feed problem is a very demanding one. These are just two examples of the myriad of possible search spaces.

7.2.3 Tree antennas

Another major variation involves using a completely different kind of chromosome to generate an antenna: a tree chromosome. Tree antennas are thus very different from all previous antenna structures. The tree chromosome can have 3-D coordinates in its nodes. Wires for the NEC2 input file are drawn between parents and children in the tree. The chromosome is actually a fractal structure, able to branch, and very large trees with very complex connections can result. This freedom has led to some surprising preliminary results, but some disappointing ones as well.

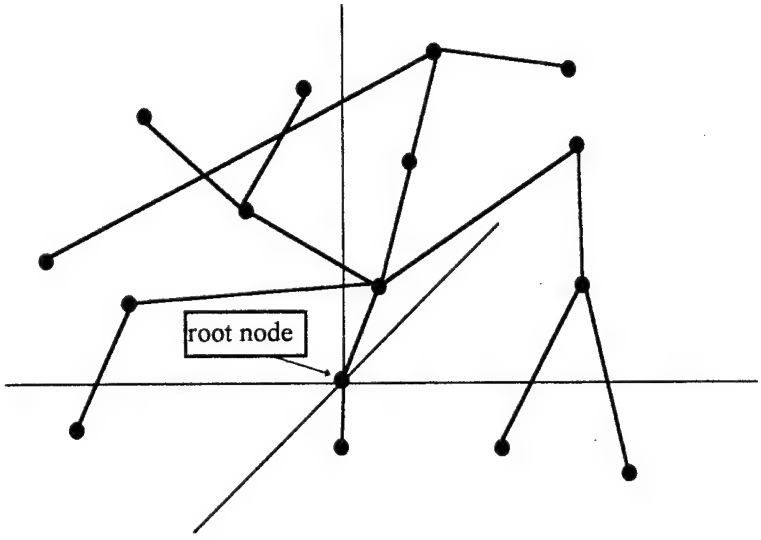


Figure 7.6. Tree Antenna Design Space. The tree has 3-D point coordinates in its nodes. Wires are connected between parent and child nodes. The root node is at the origin. Each wire can be a designated size, allowing for only angular data to reside in the nodes, or wires can be of different distances.

For the preliminary runs, each wire in the tree was limited to one segment in length (0.075λ), and each node contained relative coordinate information from its parent—i.e., its direction relative to its parent node. The program that writes the NEC2 input file takes this relative information and converts it to absolute coordinates, as NEC2 requires.

The initializer routine for the tree constructed a tree with a certain maximum number of children per node, and a certain maximum depth. It places random 3-D vector information in each node, where the vector is 0.075λ long and points in a random direction. The mating process included both node and subtree swapping, and mutation included subtree destruction, node replacement and swapping, and subtree swapping.

Following are two resulting antennas. They were applied to the Arecibo Feed problem, so they were optimized for a 60° beamwidth centered at 0° θ, low sidelobes, and linear polarization. To facilitate the linear polarization, they were limited in movement in the Y plane, producing almost flat antennas. They were allowed some movement in this third dimension to allow fewer wire crossings. They were both generated over a ground plane.

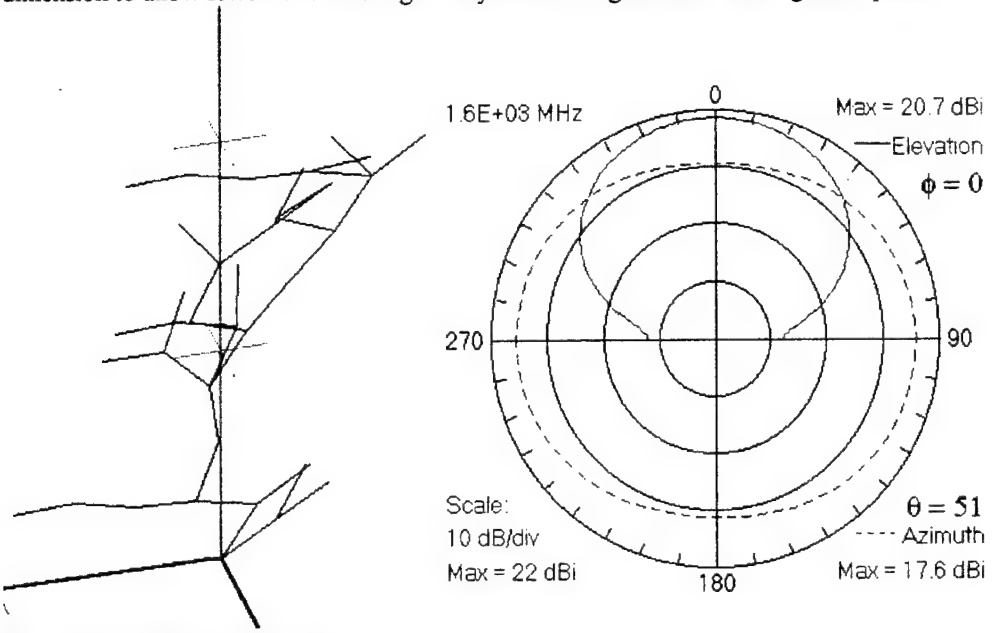


Figure 7.7. Preliminary Result 1.

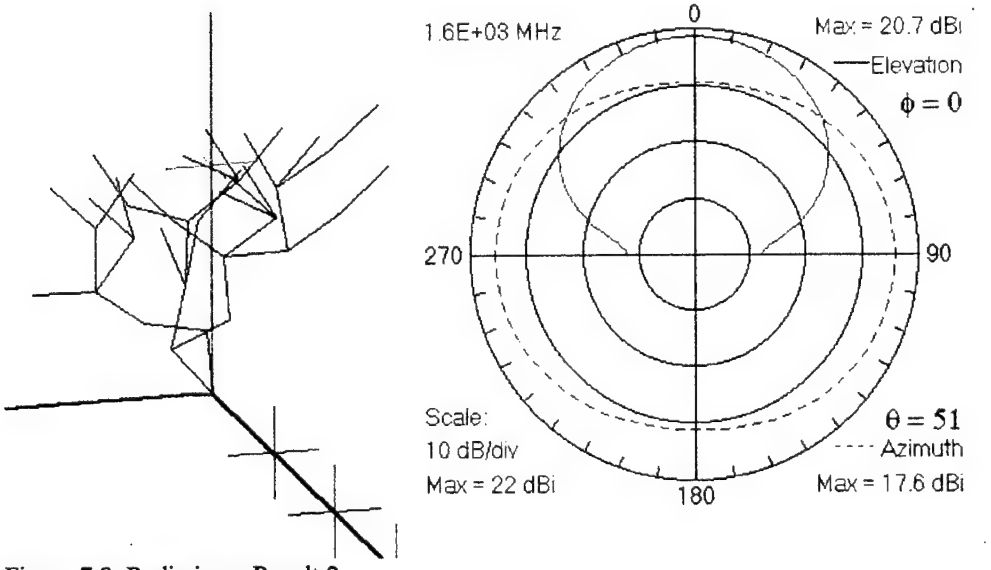


Figure 7.8. Preliminary Result 2.

Notice that they seem to have very regular-looking patterns for such unusual shapes. It should be stated that it is almost certain that these two antennas violate the assumption in NEC2 that wires are not touching except at segment boundaries. This leads to the high absolute gains that are shown, which are in all likelihood false. However, the antenna directivity pattern is generally robust to such violations, and shows a well-behaved pattern is possible with these chaotic-looking structures. It was also easy to see that these structures were too cumbersome to put into the

rigorous environment of the Arecibo antenna. However, these results did indicate that much could be done with this antenna type.

Since the preliminary results looked so promising, an experiment was run with the tree antenna GA, as applied to the RH circular polarization, hemispherical problem of the crooked wire antenna.

The 72-run experiment varied maximum number of wires (10, 18, 25), absolute or relative references in the nodes (i.e., whether nodes contained points that specified absolute coordinates, or relative distances from the parent node), overlap (35%, 65%), mutation rate (0.6%, 0.06%), number of levels/layers allowed in the trees of the first generation (2, 3, 4). This last parameter determines the depth of the tree. For instance, tree of preliminary result 2 above has six layers. This parameter only affects the first generation; future generations can acquire more levels from subtree swapping. Held constant was the number of children allowed for each parent node, set at 4. The mating process included both node and subtree swapping, and mutation included subtree destruction, node replacement and swapping, and subtree swapping. Wires were allowed to vary in length, as opposed to the preliminary runs.

The results of the experiment showed the best individuals should be obtained from the following combination: 10 nodes maximum, absolute referencing, 35% overlap, 0.2% mutation, 4 levels in the initial generation. The predicted number of NEC2 simulations was 3489 and the score was 1173. Confirmation runs averaged 3425 NEC2 simulations each, as predicted, but had scores of only 2040 on average—much poorer than predicted. Thus, more work remains to be done on this type of antenna.

The preliminary results, then, seem to hold more promise than was realized thus far. The tree antenna needs more work to make it useful and possible to fabricate. Notice the jumbled appearance of the two preliminary results—actually constructing that antenna would be very difficult. Restrictions probably should be placed on the structures to ensure they do not violate the assumptions of the simulator (as they almost certainly have done) and they are able to be easily assembled. More work also needs to be done to discover the types of problems best solved with these structures, and the best parameters for the GA that optimizes them.

7.2.4 Using developmental rules

Another particularly intriguing idea comes from the fact that biological life is not specified by the genetic code in the way that antenna designs have thus far been specified. The shape and color of a leaf or a flower is determined, not by some external builder that sees a gene for a certain shape and builds that shape, but by hundreds of thousands of cells, each working from a complete set of the genetic code and specializing in different ways to form the different structures that make up the anatomy of a plant or animal. Each cell works from a set of rules for specialization and function, that collectively make the structure grow and live and have its characteristic shapes or colors.

It would be interesting to make antennas using rules rather than specifications, and use a GA to evolve, not the antenna itself, but the rules that tell one when to use, for instance, a single metal segment, or a jointed metal segment. A set of rules could be evolved that specify when a metal trail should turn, branch or stop, given its surroundings. As another example, a group of "cells" with certain internal metal structures could be used to spawn others in a "growing" antenna, and rules could tell these cells when to branch or what direction to spawn in and when to stop growing. Again, the possibilities are endless.

7.3 Conclusion

There are almost no limits on the search spaces that are possible to optimize with a GA. Wires can be configured in almost any conceivable configuration, and NEC2 can simulate almost any arbitrary wire structure that does not violate its basic assumptions. This incredible freedom indicates almost limitless potential for antennas in the future.

There are also other metal shapes besides thin wires, of course. A fast simulator that can simulate the antenna patterns of arbitrary shapes could be used to construct genetic antennas made of plates or other solid objects instead of, or in addition to, wires, expanding the universe of possible search spaces dramatically.

This chapter was provided in the hopes that those who read it will be encouraged to try new and creative methods of enhancing the GA process and search spaces for different cost functions. The genetic antenna is a new area of research that promises to be a fascinating and useful addition to the field of electromagnetic structures.

Chapter 8: Conclusion

It is hoped that the reader now has an understanding of the GA and how it can be applied to electromagnetic problems. After introducing the reader to antenna design problems and methods in Chapters 1 and the properties of antennas in Chapter 2, Chapter 3 covered the setup and running of a GA, with a simple two-variable antenna example that shows how a GA operates and what can be expected during typical runs. The example also shows that even simple electromagnetic problems can have complicated search spaces, and that the GA is capable of finding excellent answers in them.

Three different antennas were then discussed in detail: the loaded monopole, the Yagi antenna, and the crooked-wire genetic antenna. The GA found an asymmetric loaded monopole with an average variation in gain over the hemisphere of only 0.4dB, and this antenna was built and measurement confirmed this result. But not only was the loaded monopole optimized by the GA, but the loaded monopole search space was explored and the GA itself was optimized for this problem. Because there were only six unknowns, the search space was exhaustively explored and was found to be complicated and multi-modal. Both real and binary versions of the GA were used to optimize this antenna, and the real GA was found to require no parameter-tuning specific to this antenna while producing results nearly identical to a tuned binary GA in about 25% of the NEC2 simulations.

The Yagi antenna was studied in Chapter 5, and GA-optimized Yagi antennas surpassed the gain of conventional Yagis by about 1dB, improvement also confirmed by measurement. The GA designed a Yagi with a beamwidth of 50°-60°, sidelobes nearly 25dB down, and a 14% bandwidth—specifications difficult to achieve using conventional techniques. The real GA was found to produce better results with this antenna over the binary GA given a comparable number of NEC2 simulations.

The crooked-wire genetic antenna, the most unusual antenna of the three, was discussed in Chapter 6. Optimization for hemispherical coverage with right-hand circular polarization (RHCP) produced highly unusual shapes unrealizable using a conventional approach. RHCP hemispherical coverage was achieved with less than 4dB variation. Measurements verified the results for two such antennas. In addition, the search space was studied, and it was found that a random chromosome would produce a design with a fitness that fell within a slightly skewed normal distribution and the average random antenna was poor, but not exceedingly so. This distribution showed that it requires good luck or skill to produce an antenna that is able to perform well or perform extremely poorly. In addition, the binary GA was tuned with experiments for the optimization of this antenna, allowing good designs to be produced from about 50% of the GA runs, which required about 3750 NEC2 simulations each. However, the real GA, with no tuning, was able to find a better design than had ever been found in all binary GA runs, tuned or not, in only one run and about 25,000 NEC2 simulations.

As shown in Chapter 7, there are many avenues of further research, in two main areas: improvement of the GA and the exploration of unusual antennas. Response-surface modeling and using series of GAs with increasing resolution show great promise in making the GA more efficient, while multi-chromosomal and tree antennas show promise at being able to solve difficult design problems.

Using the GA, it appears that few antenna problems are insurmountable, given enough time and computer power. Those that are still intractable are so because their simulations require too much time and/or memory. But as computers become more powerful, and simulators become more streamlined, even these problems should fall within the range of the GA optimization technique.

While much progress remains to be made, the GA looks extremely promising as a technique that can open the door to many different optimizations. There may be faster optimization methods for certain problems, and there may be more powerful techniques discovered in the future, but right now the GA appears to be the most general and the most powerful optimization technique for the spiky, complicated search spaces that antennas have. It is hoped that the GA will further the boundary of what is possible with antenna structures, and relieve the engineer from tedious and difficult optimization by hand, allowing him or her to be faster, more creative, and more effective at solving the problems facing the world of communications and remote sensing.

Bibliography

[1] E.E. Altshuler, "A monopole antenna loaded with a modified folded dipole," IEEE Trans. Antennas Propagat. Vol. 41, pp. 871-876, July 1993.

[2] E.E. Altshuler, "Hemispherical coverage using a double-folded monopole," IEEE Trans. Antennas and Propagat., Vol. 44, August 1996.

[3] E.E. Altshuler, "A monopole loaded with a loop antenna," IEEE Trans. Antennas and Propagat., Vol. 44, pp. 787-791, June 1996.

[4] D. S. Weile and E. Michielssen, "Genetic algorithm optimization applied to electromagnetics: a review," IEEE Trans. Antennas Propagat., vol. 45, pp. 343-353, March 1997.

[5] R.L. Haupt, "Thinned arrays using genetic algorithms." IEEE Trans. Antennas and Propagation, Vol. 42, July, 1994, pp. 983-999.

[6] R.L. Haupt, "An introduction to genetic algorithms for electromagnetics," IEEE Ant. and Propagat. Magazine, Vol. 37, pp. 8-15, April 1995.

[7] A. Cantoni, et al. "A New Approach to the Optimization of Robust Antenna-Array Processors" IEEE Trans. on Ant. and Propagation, Vol. 41, Iss 4, 1993, pp. 403-411.

[8] E. Michielssen, et al. "Design of electrically loaded wire antennas using massively parallel genetic algorithms," in Proc. URSI Radio Sci. Meet., Seattle, WA, June 1994.

[9] D.S. Weile, et al. "Optimization of broad-band loaded wire antennas in real environments using genetic algorithms," in Proc. URSI Radio Sci. Meet., Seattle, WA, June 1994, pp. 726-733.

[10] Z. Altman, et al. "New designs of ultra-broadband antennas using genetic algorithms," in Proc. IEEE Antenna Propagation Soc. Int. Symp., Seattle, WA, June 1994, pp. 2054-2057.

[11] M. Bahr, et al. "Design of ultra-broadband monopoles," in Proc. IEEE AP-S Symposium, Seattle, WA June 1994, pp. 1290-1293.

[12] A. Boag, et al. "Design of electrically loaded wire antennas using genetic algorithms," IEEE Trans. Ant. and Propagat., vol. 44, pp. 687-695, May 1996.

[13] M..J. Johnson, and Y. Rahmat-samii, "Genetic algorithm optimization and its application to antenna design." Proceedings of the IEEE AP-S Symposium, Seattle, WA, June 1994.

[14] M..J. Johnson, and Y. Rahmat-samii, "Genetic algorithms in electromagnetics," Proceedings of the IEEE AP-S International Symposium, Baltimore, MD, July 1996.

[15] D.S. Linden and E.E. Altshuler, "Automating Wire Antenna Design using Genetic Algorithms," Microwave Journal, Vol. 39, pp. 74-86, March 1996.

[16] E. E. Altshuler and D.S. Linden, "Design of a loaded monopole having hemispherical coverage using a genetic algorithm," IEEE Trans. Antennas Propagat., Vol. 45, pp. 1-4, Jan. 1997.

[17] D.S. Linden and E.E. Altshuler. "The design of Yagi antennas using a genetic algorithm," in Proceedings of the USNC/URSI Radio Sci. Meeting, Baltimore, MD, July 1996, p. 283.

[18] G.J. Burke and A.J. Poffio, "Numerical Electromagnetics Code (NEC)-Method of moments, " Rep. UCID18834, Lawrence Livermore Lab. CA, Jan 1981.

[19] L.J. Chu, "Physical limitations of omni-directional antennas," Journal of Applied Physics, Vol. 19, pp. 1163-1175, December 1948.

[20] David H. Staelin, et al. 6.014: Electromagnetic Waves (course notes). MIT, May, 1992.

[21] J.H. Holland, "Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975).

[22] David E. Goldberg, *Genetic Algorithms in Search, Optimization & Machine Learning*, Addison-Wesley (1989).

[23] Anthony J. Griffiths, et al. *An Introduction to Genetic Analysis*, (5^{th} ed.) W.H. Freeman and Company, New York (1993).

[24] A. Adewuya, " New Methods in Genetic Search with Real-valued Chromosomes," Master's Thesis, Mech. Engr. Dept., MIT, 1996

[25] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, (2^{nd} ed.) New York: Springer-Verlag (1994).

[26] ARRL Handbook, Chapter 20, 1996.

[27] J. Reisert, Personal communication, 1996.

[28] J. Weintroub, Personal Communication ,1996.

[29] J. Greffenstette, "Virtual Genetic Algorithms: First Results," to be published.

[30] K.A. De Jong, "An analysis of the behavior of a class of genetic adaptive systems." Doctoral Dissertation, Univ. of Michigan. *Dissertation Abstracts International* 36(10), 5140B. (1975) (University Microfilms No. 76-9381)

[31] F.M. Landstorfer and R.R. Sacher, *Optimisation of Wire Antennas.* New York: John Wiley and Sons Inc. (1985).

[32] C. Chapman, K. Saitou,, M. Jakiela, "Genetic Algorithms as an Approach to Configuration and Topology Design," ASME Journal of Mechanical Design, Vol.116, December 1994, pp.1005-1012.

[33] Martin S. Smith, *Introduction to Antennas*, London: Macmillan Education, Ltd. (1988).

[34] W.L. Weeks, *Antenna Engineering*, New York: McGraw-Hill, Inc. (1968).

[35] D.R. Wallace, *A Probabilistic Specification-based Design Model: applications to design search and environmental computer-aided design*, Ph.D. Thesis, Mech. Engr. Dept., MIT, 1994.

Appendix A: NEC2 Limits and Validation

NEC2 uses a method-of-moments calculation to achieve antenna patterns and characteristics. This method requires that the simulator break an antenna design into small, finite segments, which it then approximates as a single point in the simulation. This technique has been well-validated and used with great success over many years.

Because there are finite elements approximating the actual design, care must be taken to follow certain rules to ensure the answers are valid. Among other assumptions, NEC2 uses a thin-wire approximation, which assumes that the radius of the wire is much smaller than the length of the segment. Thus, if one attempts to include too many segments in the hopes of creating a more accurate simulation, one will violate this rule and actually decrease the simulation accuracy.

The rules that one must follow, then, are contained in the next section.

A.1 Limitations and tests to ensure validity

Following is a list of NEC2 limitations and litmus tests to ensure that results are valid, compiled from the literature listed in the next section:

I. Wires and segmentation
- A. Segments should be no longer than $\lambda/10$ (Burke et al.). $\lambda/5$ is an upper limit for weak, smoothly varying regions of current found by Peng, et al.
- B. Segments must not be so small they violate thin wire assumption: length/radius > 2 for 1% accuracy, and > 0.5 for reasonable results. (Burke et al.)
- C. Peng, et al. suggest that the lower limit due to degeneracy in the basis functions of NEC is $\lambda/1000$, and problems with circulating currents may occur with segments shorter than $\lambda/250$.
- D. Intersecting segments are not connected unless they do so at their ends--large errors can arise if the evaluation point for one segment lies within another.
- E. Wire radius must be small: 2π radius $/\lambda << 1$ (Burke).
- F. Simulated wire radii should be close to physical ones (Peng).
- G. Close parallel wires should have identical segmentation (Peng).

II. Impedances are difficult to calculate accurately—care must be taken near source for accurate impedance calculation, whereas radiation pattern is still accurate without such care (Austin and Fourie).

III. Finite ground planes
- A. Surface area of wires should be equal to surface area of ground plane (Peng, et al.).
- B. Wires should be no farther than $\lambda/10$ apart (Peng, et al.).

A.2 Literature review

As mentioned previously, NEC2 has been a standard in the antenna community for years. As such, it has been used in many different types of wire antenna research. Table I describes published papers that show close agreement between NEC2 and measurements or theory.

Paper	Significance
Richie, J.E. and Gangl, H.R. III. "EFIE-MFIE Hybrid Simulation using NEC:VSWR for the WISP Experiment." IEEE Trans. on Electromag. Compat., Vol 37, No. 2, May 1995. pp. 293-296.	Shows agreement between VSWR calculations of space shuttle cargo bay with measurements.
J. Moore and M.A. West. "Simplified analysis of coated wire	Shows excellent agreement between measurement

antennas and scatterers." IEE Proc.-Microw. Antennas Propag., Vol. 142, No. 1, February 1995.	and simulation of coated wire dipole current distributions, taking coating into account through a radius change and impedance loading
Mann, S.M. and Marvin, A.C. "Characteristics of the Skeletal Biconical Antenna as Used for EMC Applications." IEEE Trans. on Electromag. Compat. Vol 36, No. 4, November 1994.	Shows close agreement between measured and predicted impedance measurements (both real and imaginary) vs. frequency, with segment size $\lambda/20$.
James, J.R. and Andrasic, G. "Environmental coupling loss effects in superconducting HF loop antenna design." IEE Proc.-Microw. Antennas Propag., Vol 141, No. 2, April 1994.	Shows close agreement between design formulas and NEC calculations of radiation resistance, reactance and efficiency for various square loop antennas.
Upton, M.E.G. and Marvin, A.C. "The Fields Due to a Small Loaded Loop in Free Space." IEEE Trans on Electromag. Compat., Vol 36, No. 1, February 1994.	Shows close agreement between theoretical analytical calculations and NEC calculations for the wave impedance of loaded and unloaded loops in free space. Also showed close agreement between measurements and NEC calculations.
Altshuler, E.E. "A Monopole Antenna Loaded with a Modified Folded Dipole." IEEE Trans. on Antennas and Propagation, Vol. 41, No. 7, July 1993. pp. 871-876.	Shows reasonable agreement in far-field antenna pattern, frequency dependence, and impedance for loaded monopole.
Peng, J., Balanis, C.A, Barber, G.C. "NEC and ESP codes: Guidelines, Limitations, and EMC Applications." IEEE Trans. on Electromag. Compat., Vol. 35, No. 2, May 1993.	Shows close agreement between NEC and ESP simulations subject to certain constraints and explores the limitations of both codes.
Cox, J.W.R. "Corroboration of a moment-method calculation of the maximum mutual coupling between two HF antennas mounted on a helicopter." IEE Proceedings-H, Vol. 140, No. 2, April 1993. pp. 113-120.	Shows agreement between NEC and circuit-equivalent model of mutual coupling between two loop antennas mounted on a helicopter. Problem required both near and far-field terms to be accurately calculated by NEC.
Recrosio, N., Fine, G., and Helier, M. "Analysis of Radiation Characteristics of Distribution Line Carriers with the NEC Code." IEEE Trans. on Electromag. Compat., Vol 35, No. 1, February 1993.	Shows good agreement between NEC calculations and measurements of fields from a medium-voltage distribution line under various configurations.
Austin, B.A., and Fourie, A.P.C. "Characteristics of the Wire Biconical Antenna Used for EMC Measurements." IEEE Trans. on Electromag. Compat., Vol. 33, No. 3, August 1991.	Shows good agreement between NEC and measurements of the far-field E field, impedance, antenna factor, gain and mismatch loss of the wire biconical antenna without a ground plane.
Baldwin, P.J., Boswell, A.G.P., Brewster, D.C., Allwright, J.S. "Iterative calculation of ship-borne HF antenna performance." IEE Proceedings-H, Vol 138, No. 2, April 1991.	Showed close agreement between NEC and measured far-field gain patterns (both magnitude and phase), input resistance and reactance, and radar cross sections for ship-borne antennas.

Table I. Papers that have compared NEC2 calculations to measurements or other simulators.

A.3 Validation of NEC2

Many antennas in this thesis were built and measured. These tests showed reasonable agreement with the NEC2 results, as is shown in Chapters 4, 5, and 6. As discussed in Chapter 4, when these tests included a ground plane, a significant amount of ripple was seen in the measured pattern that was not predicted in the simulation, due to the 1.2m x 1.2m ground plane that was used in measurement, as opposed to the infinite ground plane that was used in simulation. The ripples in the pattern arise from reflections from the edges of the ground plane. NEC2 can simulate antennas over an infinite ground screen in a few seconds, while it takes over 24 hours to simulate an accurate version of the 1.2m finite ground screen at 1600 MHz. Hence the infinite ground screen was used in the optimizations, even though the performance was somewhat different in real measurement.

Figure A.1 is a graph showing a comparison of NEC2 simulation of a crooked-wire genetic antenna with an infinite ground plane, a finite ground plane, and the actual measurement of the antenna, repeated from Chapter 6 for completeness. There is approximately a 5 dB variation in the field above an elevation angle of 10° as compared to the computed variation of about 1 dB over an infinite ground screen. On the other hand, the simulation with a finite, perfectly conducting ground screen shows very good agreement with measurement results. Notice that the peaks and valleys in both the measured data and the finite ground plane simulation agree well in location, though the simulation shows greater variation in amplitude. This discrepancy is minor, however, and is a result of the approximation of the real test conditions in the simulator. Though simulation with a finite ground screen is closer to reality, the antenna was not optimized using a finite ground screen because the GA requires several thousand such runs to produce a design, and, as mentioned above, the time to simulate the finite ground screen was prohibitive.

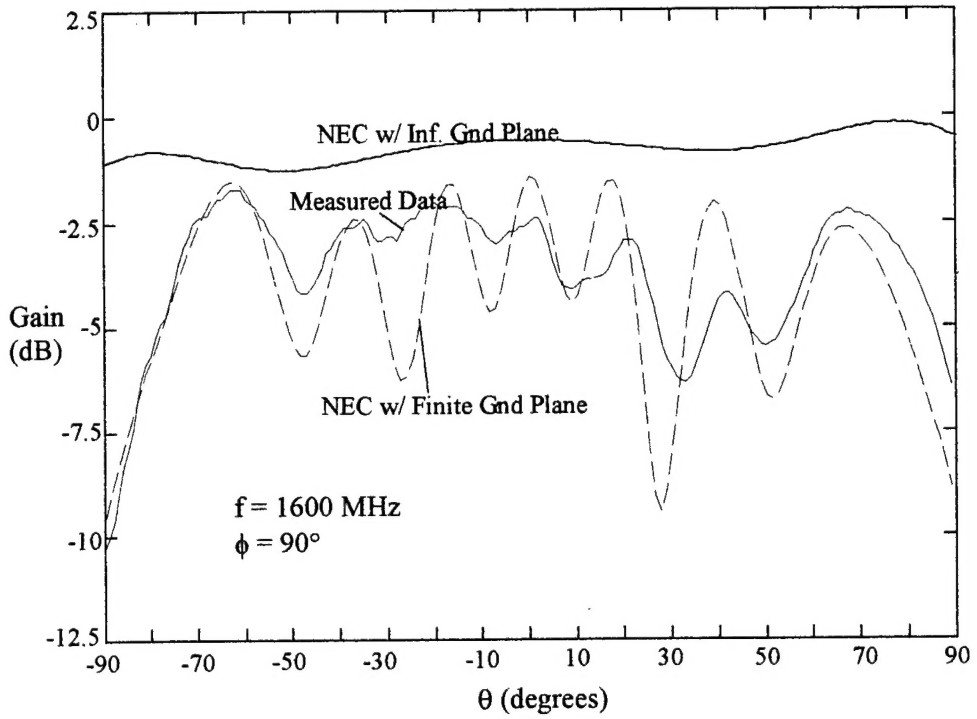


Figure A.1. Effect of the finite ground plane on an antenna pattern.

Appendix B: 2-D, 3-D, and List Chromosomes

Should the reader desire to explore these chromosomes, their structure and operators are listed here.

2-D, 3-D Chromosomes

Works best for 2-D or 3-D finite element problems

Example: structure with structural material or space in each square of a grid

Phenotype:

X		X			X		X
				X	X	X	X
		X		X			
X	X	X	X	X	X		X
X	X		X		X	X	
X		X			X		
X		X	X			X	X
	X	X	X	X	X		

Genotype:

1 0 1 0 0 1 0 1
0 0 0 0 1 1 1 1
0 0 1 0 1 0 0 0
1 1 1 1 1 1 0 1
1 1 0 1 0 1 1 0
1 0 1 0 0 1 0 0
1 0 1 1 0 0 1 1
0 1 1 1 1 1 0 0

2-D, 3-D Mating and Mutation

Parents:

1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0

Crossover:

1 1 1 1 1 | 1 1 1 0 0 0 0 0 | 0 0 0
1 1 1 1 1 | 1 1 1 0 0 0 0 0 | 0 0 0
1 1 1 1 1 | 1 1 1 0 0 0 0 0 | 0 0 0
1 1 1 1 1 | 1 1 1 0 0 0 0 0 | 0 0 0

Children:

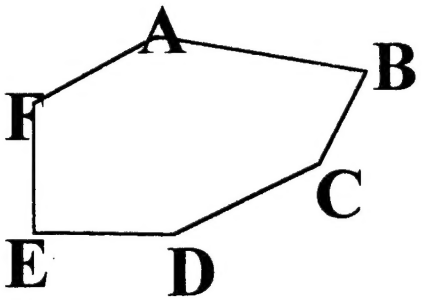
1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1
1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1
0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0
0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0

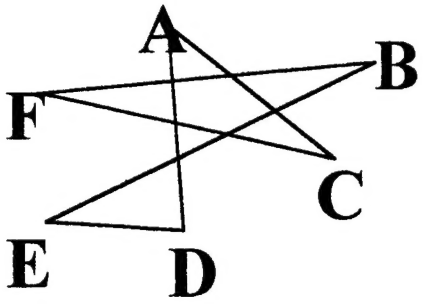
Mutation: still just a bit flip or one of the real number mutations

List Chromosomes

Best for lists with non-repeating elements

Classic list problem: the traveling salesman





ABCDEF

ACFBED

List Chromosome Mating and Mutation

ABCDEF → crossover, 3rd position → ABCBED
ACFBED ACFDEF
unfeasible children

ABCDEF → crossover, 3rd position → ABCFED
ACFBED ACFBDE

Mutation: Move a random gene to a random spot
ABCFED → AEBCFD

Appendix C: Other Mating Operators for Real 1-D Chromosomes

(All of these techniques are from [25])

Whole Arithmetic
Each of the child's genes is copied from one of its two parents at random.

Simple
Beyond the crossover point, each child gene is a linear combination of the parents' genes, taken gene by gene:
$C_i = a \cdot P1_i + (1-a) \cdot P2_i$, $0 < a < 1$

Linear
$C1_i = (0.5)P1_i + (0.5)P2_i$, $C2_i = (1.5)P1_i - (0.5)P2_i$,
$C3_i = -(0.5)P1_i + (1.5)P2_i$

Average
$C_i = [P1_i + P2_i] / 2$

BLX-α
Child is chosen uniformly from entire interval I + 2αI